# *MetrIntPair*—A Novel Accurate Metric for the Comparison of Two Cooperative Multiagent Systems Intelligence Based on Paired Intelligence Measurements

Laszlo Barna Iantovics,[1,*] Corina Rotar,[2,†] Muaz A. Niazi[3,‡]
[1]*Petru Maior University, Nicolae Iorga 1, Tg. Mures, 540088, Mures County, Romania*
[2]*"1 Decembrie 1918" University, Alba Iulia, Romania*
[3]*COMSATS Institute of IT, Islamabad, Pakistan*

In this paper, we propose a novel metric called *MetrIntPair* (*Metric for Pairwise Intelligence Comparison of Agent-Based Systems*) for comparison of two cooperative multiagent systems problem-solving intelligence. *MetrIntPair* is able to make an accurate comparison by taking into consideration the variability in intelligence in problem-solving. The metric could treat the outlier intelligence indicators, intelligence measures that are statistically different from those others. For evaluation of the proposed metric, we realized a case study for two cooperative multiagent systems applied for solving a class of NP-hard problems. The results of the case study proved that the small difference in the measured intelligence of the multiagent systems is the consequence of the variability. There is no statistical difference between the intelligence quotients/level of the multiagent systems. Both multiagent systems should be classified in the same intelligence class.


## 1. INTRODUCTION

The agent-based systems represent a relatively new field of Artificial Intelligence appropriate for many problem-solving. In our approach, we will refer to agent-based systems, agents, and cooperative multiagent systems composed of two or more agents situated in the same environment that cooperatively solve problems. Even in cooperating multiagent systems composed of simple agents, an increased intelligence at the systems' level could emerge if the member agents cooperate

*Author to whom all correspondence should be addressed; e-mail: barna_iantovics@yahoo.com.
†email: cory2000ro@yahoo.com
‡email: muaz.niazi@ieee.org

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 00, 1–24 (2017)

View this article online at wileyonlinelibrary.com. • DOI 10.1002/int.21903

efficiently and flexibly. The investigation of aspects that are related to intelligence is important in order to design highly efficient problem-solving solutions. Intelligent agent-based systems are used for many difficult problem-solving.[1–7]

In many studies, the agent-based systems' intelligence is established based on some intuitive considerations that are frequently related to the intelligence (learning and adaptation) of biological life forms (humans, animals, and swarms of insects). The intelligence estimation based on some intuitive intelligence considerations, as is realized in most of the studies, is not enough. We consider that innovative metrics are required for measuring the agent-based systems' intelligence. There are very few metrics that are able to measure the intelligence quotient of an agent-based system. There are even fewer that are able to make an accurate comparison of two agent-based systems' intelligence. Another disadvantage of most of the designed metrics presented in the literature is the limited universality. The same problem could be solved by agent-based systems with very different architecture. Different studies related to the machine intelligence quotient (MIQ) measuring were presented in papers.[8–11]

There are many studies related to the intelligence of humans.[12–16] Based on a comprehensive study of the scientific literature, we can conclude that a universal understanding of the human intelligence does not exist. There is no possibility of a full modeling of the human brain based on its enormous complexity. We considered that this is somehow similar to the very complex intelligent agent-based systems, which are much less complex than the humans, but are by a large variety. We considered appropriate the design of metrics that use problem-solving intelligence evaluations, which should be similar to the human intelligence measuring, based on different problem-solving intelligence evaluations.

In the following part of the paper, we will focus on cooperative multiagent systems intelligence. We propose a novel metric called *MetrIntPair* (*Metric for Pairwise Intelligence Comparison of Agent-Based Systems*) that is able to make an accurate comparison of the problem-solving intelligence of two cooperative multiagent systems. *MetrIntPair* takes into consideration the variability in the intelligence of the compared multiagent systems. A multiagent system could manifest different intelligence level (quotient) during different problem-solving. Two multiagent systems, belonging to the same class, with the same measured intelligence using *MetrIntPair* can be considered. The metric allows the selection of the cooperative multiagent systems based on the problem-solving intelligence. In the conclusions section, we will outline the universality of the proposed metric.

For attesting the effectiveness of the proposed metric, we formulated an illustrative case study. We considered two simple cooperative multiagent systems that are specialized in solving a class of NP-hard problems, specifically the *TSP* (*Traveling Salesman Problem*).[17–21] The two cooperative multiagent systems performed as *Rank-Based Ant System*[22,23] and *Min-Max Ant System* (*MMAS*),[23,24] both are well known in the scientific literature.

The upcoming part of the paper is organized as follows: Section 2 analyzes the intelligence of cooperative multiagent systems. There are some metrics presented for the intelligence measuring. In Section 3, the novel metric *MetrIntPair* for intelligence comparison of two cooperative multiagent systems is presented. For validation

purposes, a case study is presented in Section 4. Section 5 discusses on the proposed metric. In Section 6, the conclusions of the research are presented.

## 2. PROBLEM-SOLVING INTELLIGENCE OF THE AGENT-BASED SYSTEMS

### 2.1. Definitions of Intelligent Agent-Based Systems

Various studies are focused on the development of intelligent agent-based systems,[25–27] intelligent agents, and intelligent cooperative multiagent systems. The agent-based systems' intelligence could not be unanimously defined.[25] Mostly, the intelligence estimation is realized based on some considerations, such as[27] autonomous learning, self-adaptation, and long-term evolution. Many considerations are inspired by different life forms that evolved during more generations, able to learn during their life cycle, and self-adapt to the environment.

Dastani and Meyer[28] analyzed the role of emotions in the design of agents. They analyzed four types of emotions: fear, happiness, sadness, and anger. These emotions influence the agents' goals and the elaboration of plans. In the case of studied agents, the emotions result during the deliberation process, and they influence the deliberation process.

Contrary to the static agents, the mobile agents are able to move in the environment during the problem-solving process. Mobile agents could be classified as software mobile agents and robotic mobile agents. A software agent operates in a software environment, a computer or a computer network. A mobile robotic agent operates in a physical environment. For instance, we mention a swarm of mobile robots that are specialized in collecting objects in a physical environment.

To illustrate the impossibility to define the agents' intelligence, let us consider the differences in intelligence between static software agents versus mobile software agents.[25] Many mobile agents are more limited in intelligence than the static counterparts. Limitations in the mobile agents' intelligence are based on some practical reasons. The endowment of a software agent with intelligence may increase the agent's size in terms of code length. Therefore, an intelligent agent (having a more complex software code, which involves more intense computations) that operates at a host requires more computational resources. The transmission of a large number of intelligent mobile agents in a network might overload the network with data transmission (based on the increased size of the agents). A large number of intelligent mobile agents, which execute complex computations at a host, may overload that host with computations. The mobile agents migrate during their operation in a network. Based on this fact, it is difficult to locate where a mobile agent is at a specific moment of time. This limitation makes difficult the endowment of mobile agents with communication capability. The communication is an essential property necessary in the cooperation.

Irani and Rabani[29] analyzed the efficiency of cooperative multiagent systems in problem-solving. In the study, the load balancing and virtual circuit routing optimization problems were considered. The authors considered the problem input division among the agents. Irani and Rabani discussed the following main questions:

"Given a bound on the maximum out-degree in a graph, which is the best graph?" and "What is the quality of the solution obtained as a function of the maximum out-degree?".

Frequently, the scientific literature presents a cooperative multiagent system as being intelligent based on the simple consideration that the efficient and flexible cooperation between the member agents emerges in improvement in problem-solving. Based on this principle, in a cooperative multiagent system, the intelligence could be considered at the systems' level.[25,27] The intelligence in these systems is higher than the individual member agents' intelligence. Even in very simple cooperative multiagent systems, intelligence could emerge at the systems' level. By efficient and flexible cooperation, simple agents could intelligently solve difficult problems.

The study in Ref. 30 presented an intelligent mobile multiagent system composed of simple reactive agents. The mobile agents were specialized in a computer network administration. They were endowed with knowledge retained as a set of rules, which described administration tasks. The multiagent system could be considered intelligent based on the fact that it simulates the behavior of a human network administrator.

There are many researches[31,32] focused on the study of decision taking in the frame of cooperative coalitions. Decisions taken in the frame of coalitions often outperform the decisions of individuals who operate in isolation.

Langley et al.[26] examined the motivations for research on cognitive architectures. In the above-mentioned paper, the cognitive systems architecture, described in the scientific literature, were reviewed. The authors discussed some open issues that should drive the research related to the architecture of cognitive systems. They considered some important capabilities that a cognitive architecture should support, which include: organization, performance, and learning, and some theoretical criteria for making an evaluation at the system's level.

This section summarized some aspects related to the intelligence of agent-based systems. Designing intelligent agent-based systems represents very important research directions for efficient solving of a large variety of problems. Intelligent agent-based systems are used for many problem-solving.[1,3,4,7,33,34] There are many intelligent computing systems that are agent-based intelligent systems based on their properties but are not called agent-based systems. The large usage of intelligent agent-based system motivates the necessity of researches related to the designing of metrics for measuring their intelligence.

### 2.2. Metrics for Measuring the Intelligence of Agent-Based Systems

The existence of some properties of the cooperative multiagent systems that could be associated with the intelligence does not allow a quantitative evaluation; they just formally prove its existence. We consider that the evaluation of a system's intelligence must be based on some metrics that allow the effective measuring of "quantity of intelligence"-intelligence quotient/level and comparing a system's intelligence with the intelligence of another system.

There are some metrics that are designed for making different kinds of mea-

surements in systems that are considered intelligent. Such metrics are not always designed for measuring the system's intelligence as a whole, but for measuring some aspects that are of most interest. A fault-tolerant system is able to diagnose and recover from some faults. Sometimes this property of the systems is associated with the intelligence. Kannan and Parker[9] analyzed the fault-tolerant systems in a comprehensive study. Kannan and Parker[9] proposed an effective metric for the evaluation of fault tolerance.

Schreiner[8] presented a study realized by the US National Institute of Standards and Technology (NIST). The study was related to creating standard measures for systems that can be considered intelligent. Schreiner[8] accentuated the question related to how precisely intelligent systems can be defined and how to measure and compare the capabilities that intelligent systems should provide. NIST's initial approach for establishing metrics attempts to address different theoretical and pragmatic subjects.

Fox et al.[35] presented a research focused on agents with cognitive capabilities that can be considered intelligent. They considered the BDI agents (with Belief–Desire–Intention architecture) very important as a class of cognitive agents. They designed a benchmark agent model appropriate for comparing agent-based systems. PROforma is an agent technology for modeling medical expertise. The benchmark was realized in order to carry out a case study analysis of this technology. It was analyzed based on three points of view: object-oriented programming, logic programming, and agent-oriented programming. The strengths and weaknesses of PROforma were analyzed.

Chmait et al.[36] studied the intelligence of multiagent systems. They considered that the intelligence is influenced by the communication and observation abilities of the member agents. The authors formulated the research question, "Which factor has a more significant influence?". The solving of the research question was approached by using an information-theoretical approach. Using some tests collaborative agents across different communication and observation abilities were compared. Based on the obtained results, Chmait et al.[36] formulated the conclusion that the effectiveness of multiagent systems with low observation/perception abilities can be significantly improved by making the communication more efficient. They presented some situations where these assumptions were not verified, and also analyzed the dependency between the studied factors.

Anthon and Jannett[10] considered the agent-based systems' intelligence based on the ability to compare alternatives with different complexity. In their research, an agent-based distributed sensor network system was considered. For measuring the intelligence, a specific approach was applied. The proposal was tested by comparing the MIQ on different agent-based scenarios.

Hibbard[11] proposed a metric for intelligence measuring based on a hierarchy of sets of increasingly difficult environments. An agent's intelligence was measured according to the ordinal of the most difficult set of environments that it can pass. The proposed measure was defined in Turing machine and finite state machine models of computing. In the finite model, the measure includes the number of time steps required to pass the test.

Chmait et al.[37] presented an interesting study related to the intelligence of agent

coalitions/groups. It proposed a metric considered "universal" and appropriate to empirically measure intelligence for different agents. The effectiveness of various algorithms was evaluated on a per-agent basis over a selection of tasks by different complexities. The study presented different situations over which a cooperative multiagent system can be more intelligent than others. It discussed some of the factors in influencing the collective intelligence.

Legg and Hutter[38] proposed a formal measure for intelligence. The authors considered that the performance in easy environments counts less toward an agent's intelligence that does performance in difficult environments.

An interesting study related to the intelligence of computing systems was realized by Legg and Hutter.[39] They considered that a fundamental problem in AI is that the notion of intelligence cannot be uniquely defined. Based on the researchers' affirmation, "nobody really knows what intelligence is." Legg and Hutter[39] outlined that the artificial systems are significantly different from humans. In the presented research, some given definitions to the human intelligence were considered. Based on these considerations, they established some essential features. These features were mathematically formalized in order to produce a general measure of intelligence for arbitrary machines. The objective was to capture the concept of machine intelligence. In the study, a survey of other tests and definitions of the machine intelligence was also realized. We fully agree with Legg and Hutter in relation to the intelligence of computing systems that cannot be defined uniquely. The evolution of life on earth began 0.5 billion of years ago.[40] The advances in technology and computing systems (software/hardware) are very fast, but will take a very long time until the computing system will have the complexity and intelligence of the most advanced life forms on earth.

Hibbard[41] proved that a constraint on universal Turing machines is necessary for Legg's and Hutter's formal measure of intelligence to be unbiased. The measure proposed by Legg and Hutter, defined in terms of Turing machines, is adapted to finite state machines. A No Free Lunch result is proved for the finite version of the measure.

Legg and Veness[42] analyzed the formal definition of machine intelligence called Universal Intelligence Measure (UIM) based on Hutter's Universal Artificial Intelligence theory. It is an extension of Ray Solomonoff's work called universal induction.[43–45] Legg and Veness considered that the UIM is asymptotically computable, and building a practical intelligence test based on this principle is not appropriate based on applicative considerations. Legg and Veness studied some practical issues involved in developing an applicable UIM. They developed a prototype implementation that was used in order to evaluate different artificial agents.

In Ref. 46, the collective intelligence of particle swarm system was assessed according to a proposed Maturity Model. The proposed model was based on the *Maturity Model* of *C2* (*Command and Control*) operational space and the model of *Collaborating Software*. The main aim of the study was to obtain a more thorough explanation of how the intelligent behavior of the particle swarm emerges.

In Ref. 47, the idea of a general test called universal anytime intelligence

test was proposed. The authors of the study considered that such a test should be able to measure the intelligence level (any low or any high) of any biological or artificial system. It was based on the C-tests and compression-enhanced Turing tests developed in the late 1990s. The authors of the research discussed different tests by highlighting their limitations. They introduced some new ideas that were considered necessary for the development of a universal intelligence test.

Some research efforts related to measuring the machine intelligence are focused on the elaboration of intelligence testing frameworks. Chmait et al.[48] in recent studies elaborated some dynamic intelligence tests for measuring the collective intelligence. The work in Ref. 48 presented a technical description of a testing framework, design, and implementation. It was presented how it can be used to quantitatively evaluate the machine intelligence. Alternative testing environments were discussed in that study.

We would like to note here that the analysis of the necessary intelligence is an important aspect that should be treated in this topic during the development of intelligent systems. This is important because sometimes an increased intelligence may even have disadvantages. For instance, if we were to consider an extremely intelligent agent as one that uses complex specializations for processing, but solves very simple problems, it would probably have to make considerably more complex computations (e.g., verification of numerous conditions) than would actually be necessary.

In very few papers, some evaluations or analysis of the system's intelligence are discussed. Most of the metrics described in the scientific literature are designed for making different kinds of measurements in systems that are considered to be intelligent. Some metrics are not developed for measuring the systems intelligence as a whole, but only for measuring some aspects that represent interest. There are very few designed metrics that are able to make an effective comparison of more multiagent systems intelligence. Many designed metrics are dependent on some aspects, such as the member agents' architecture, the multiagent system architecture, and so on. Based on this fact, their effective practical utilization possibility is limited. An agent-based system for solving the same type of problems could have different architecture.

These considerations motivate the necessity to design metrics that are able to make an accurate comparison of the intelligence of more multiagent systems based on different problem-solving intelligence measurements. We consider that such a metric must be universal and should not depend on some relatively particular aspects like the agent-based systems' architecture. In the discussion section, we will compare our proposed metric with some other metrics presented in the scientific literature.

## 3. THE PROPOSED METRINTPAIR METRIC FOR COMPARISON OF TWO MULTIAGENT SYSTEMS INTELLIGENCE

In this section, we propose a novel metric for the accurate comparison of the problem-solving intelligence of two cooperative multiagent systems. The metric is described as an algorithm, which is further called *Multiagent Systems Intelligence*

*Comparison.* The metric is abbreviated as *MetrIntPair.*

Next, we consider two cooperative *MAS* denoted as *MAS1* and *MAS2* that can solve the same types/classes of problems. We consider the intelligence testing of *MAS1* and *MAS2* on the same set of test problems denoted $Prob = \{Prob1, Prob2, \ldots, Probn\}$. The obtained intelligence indicators, as a result of problem-solving intelligence evaluations, are denoted as $Set1 = \{A1t1, A1t2, \ldots, A1tn\}$ and $Set2 = \{A2t1, A2t2, \ldots, A2tn\}$. $|Set1|$ and $|Set2|$ represent the cardinality/sample size of *Set1* and *Set2*, where $|Set1| = |Set2| = n$. Table I presents the experimental conditions for the intelligence measuring, the problems and the corresponding measured intelligence indicator results for both *MAS*s. *A1t1*, *A1t2* . . . *A1tn* represents the intelligence of the *MAS1*; where *A1t1* represents measured intelligence in the *Prob1* solving, . . . , *A1tn* represents the measured intelligence in *Probn* solving. *A2t1*, *A2t2*, . . . , *A2tn* represents the intelligence of the *MAS2*; where *A2t1* represents the measured intelligence in the *Prob1* solving, . . . , *A2tn* represents the measured intelligence in the *Probn* solving.

**Table I.** Pairwise measuring of MAS1 and MAS2 intelligence.

| Problem used in intelligence testing | Intelligence evaluation of *MAS1* | Intelligence evaluation of *MAS2* | Formed pairs |
|---|---|---|---|
| *Prob1* | *A1t1* | *A2t1* | *A1t1-A2t1* |
| *Prob2* | *A1t2* | *A2t2* | *A1t2-A2t2* |
| . . . . . | . . . . . | . . . . . | |
| *Probn* | *A1tn* | *A2tn* | *A1tn-A2tn* |
| | *CentrInt1* of *Set1* | *CentrInt2* of *Set2* | |

*CentrInt1* and *CentrInt2* represent the central intelligence indicators of the *MAS1* and *MAS2*. We considered as the central intelligence indicators of *MAS1* and *MAS2* the means of *Set1* and *Set2; CentrInt1* = *Average*(*A1t1*, *A1t2*, . . . , *A1tn*), *CentrInt2* = *Average*(*A2t1*, *A2t2*, . . . , *A2tn*). The decision for choosing the mean as the central intelligence indicator was based on the fact that both data samples should be normally distributed, sampled from a Gaussian population. If the intelligence indicators data are not normal then a transformation can be applied.

The *MetrIntPair* metric compares the intelligence of *MAS1* and *MAS2* on a testing problem set *Prob*. It checks if the results (more concretely the central intelligence indicators) are equal or different from the statistical point of view. We call in the following, *Null Hypothesis* denoted as *H0*, the statement that *MAS1* and *MAS2* intelligence are equal from the statistical point of view. We denote *H1* the *Alternative Hypothesis,* a hypothesis which indicates that the intelligence of *MAS1* and *MAS2* is different from the statistical point of view.

### *MetrIntPair Metric: Multiagent Systems Intelligence Comparison Algorithm*

**IN**: *Set*1 = {*A1t1*, *A1t2*, . . . , *A1tn*}; *Set2* = {*A2t1*, *A2t2*, . . . , *A2tn*}
**OUT**: //The established Central Intelligence Indicators of MAS1 and MAS2.
//The established *Central Intelligence Indicator of MAS1*

*CentrInt1*, [*LowerCI1*,*UpperCI1*]; *CentrInt2*, [*LowerCI2*, *UpperCI2*];
//The established Central Intelligence Indicator of *MAS2*
*CentrInt2*, [*LowerCI2, UpperCI2*]
//The decision related with the intelligence comparison of *MAS1* and *MAS2*
*DecisionIntelligence*

**Step 1.**

@Establishes the $\alpha$_metr parameter value.
@Make a statistical characterization for *Set*1 and *Set*2.
@Calculate the *CV* (*Coefficient of Variation*) for both *Set*1 and *Set*2.
@Analyze the heterogeneity of *Set*1 and *Set*2 based on the *CV* value.
@Verify if *Set*1 and *Set*2 are normally distributed.
@If decided opt for the outlier intelligence values detection from *Set*1 and *Set*2.
***If*** (it was applied the intelligence indicators outliers detection) ***then***
@Eliminates the outliers.
@Verify again if *Set*1 and *Set*2 are normally distributed.
***EndIf***
@If *Set*1 or *Set*2 are NOT normally distributed then apply a transformation.

**Step 2.**

***If*** (both *Set*1 and *Set*2 are normally distributed) ***then***
//The central intelligence indicators are calculated as the means.
@Calculates *CentrInt1* and *CentrInt2* as the means of *Set*1 and *Set*2.
@Set ConfLev. //Set the confidence level of the confidence interval.
//establishes the confidence level of the confidence interval.
@Establishes the *ConfLev*.
//Calculate the confidence interval for the central intelligence indicators.
@Calculate *LowerCI*1, *UpperCI*1; *LowerCI*2, *UpperCI*2.
@Apply the parametric *Paired Two-Sample T-test*.
@As a test result will be obtained the *P-value*.
@Verify if the Pairing was Effective.
***EndIf***

**Step 3.**

@Report *CentrInt1*, [*LowerCI1*, *UpperCI1*]; *CentrInt2*, [*LowerCI2, UpperCI2*].

**Step 4.**

***If*** (P-*value*$>$ $\alpha$_*metr*) ***then***
@Accept *H0*.
*DecisionIntelligence* = "*MAS1 intelligence is equal with MAS2 intelligence*".
***Else***
@Accept H1. //H0 is rejected.
***If*** (*CentrInt1*$<$ *CentrInt2*) ***then***
*DecisionIntelligence* = *"MAS1 is less intelligent than MAS2*"
***Else***
*DecisionIntelligence* = *"MAS1 is more intelligent than MAS2"*
***EndIf***
***EndMultiagentSystemsIntelligenceComparison***

The metric used as input *Set1* = {*A1t1*, *A1t2*, . . . , *A1tn*} and *Set2* = {*A2t1*,

*A2t2*, ..., *A2tn*} represent the *MAS1* and *MAS2* intelligence indicators obtained during the *MASs* evaluation in solving of the test problems set *Prob*. |*Prob*| = cardinality/number of problems (intelligence evaluations). The metric algorithm could be applied to a Gaussian intelligence indicator data (*Set1* and *Set2* should be sampled from a Gaussian population) and also it should satisfy the restriction related to the effectiveness of data pairing by a correlation and regression analysis.

If the sample intelligence data do not follow a Gaussian distribution, a solution is the application of a transformation to the values in order to obtain normally distributed data. This is required for the application of the proposed metric. Some of the most common normalizing transformations based on applicative considerations are indicated in Table II.[49]

**Table II.** Transformation that can be applied to obtain normally distributed data.

| Type of data and distribution | Normalizing transformation |
|---|---|
| Count (*C* comes from Poisson distribution) | Square root of *C* |
| Proportion (*P* comes from Binomial distribution) | Arcsine of square root of *P* |
| Measurement (*M* comes from Lognormal distribution) | Log(*M*) |

*CV* should be calculated using Equation 1. In Equation 1, the Standard Deviation (SD)/Mean is multiplied with 100 for finding the result in percent (%). For example, if SD/Mean = 0.2 multiplied by 100 gives 20%.

$$CV = 100 \times (\text{SD}/\text{Mean}). \tag{1}$$

We used the *CV* value for analyzing the homogeneity/heterogeneity of the intelligence indicator data. A classification of the homogeneity–heterogeneity of the data can be realized based on the *CV*.[50,51] $CV \in [0,10]$ indicates homogeneous data; $CV \in (10,30)$ indicates relatively homogeneous data; $CV \geq 30$ indicates heterogeneous data.

In our study, we used a method for the elimination of outlier intelligence indicator values. We call outlier intelligence indicator value, a very high or very low measured intelligence value, different from the other intelligence indicator values. We consider that the difference of an intelligence indicator value from those others should be considered from the statistical point of view. Based on this fact, we determined that the application of statistical tests for outlier intelligence values detection is suitable. There are many tests for statistical outliers' detection described in the scientific literature, such as Chauvenet's criterion,[52,53] Peirce's criterion,[54] Dixon's *Q* test,[55] and Grubbs test.[56]

We suggest the application of the Grubbs test for outliers detection with the significance level $\alpha\_grubbs = 0.05$. At the first application of the Grubbs test, it is able to identify a single outlier (if there is at least one outlier). If a value is identified as an outlier, then it can be concluded that this is the most statistically different value of all other measured intelligence indicator values. If it is identified as an outlier, then a decision can be taken regarding if the outliers detection test will be applied again. This is a recursive process; the detection could be applied consecutively more times until there are no other outliers detected.

The decision related with the outliers' elimination, if outliers appear, should be taken by the human evaluator based on a pre-analysis and realized before applying the metric. Possible decisions could be:

- the outliers should be taken into consideration based on the fact that they are characteristic to the intelligence. It is expected to have a large variation in intelligence, for example, there are solved problems by a larger variety of types and/or complexity;
- the outliers should not be taken into consideration. They indicate some situations that do not characterize the intelligence. For example, the agents are overloaded with tasks and they could solve problems very slowly, and such a situation is very rare.

$\alpha\_metr$ denotes the significance levels when the statistical parametric *Paired Two Sample T-test* is applied. We suggest as the value of $\alpha\_metr = 0.05$, which we consider as the most appropriate. $\alpha\_metr$ denotes the probability to make a type I error, to reject *H0* (*Null Hypothesis*) when it is true. $Df(set1,set2) = n - 1$ denotes the degrees of freedom for the *Paired Two Sample T-test*. *ConfLev* represents the confidence level of the central intelligence indicators. In many cases, we recommend a value of 95%.

In the proposed metric algorithm, if $P\text{-}Value > \alpha\_metr$, then it can be concluded that *H0* could be accepted. Based on the decision, it can formulate the conclusion that even if there is a numerical difference between the calculated central intelligence indicators of *CentrInt1* and *CentrInt2,* there is no statistical difference between the intelligence of the two multiagent systems. The numerical difference is the result of an accident and/or the variability in the intelligence of the multiagent systems. In this situation, from the classification point of view, both multiagent systems *MAS1* and *MAS2* can be classified in the same intelligence class.

If *H1* is accepted, then it can be concluded that the intelligence level of *MAS1* is different from the intelligence level of *MAS2*. The numerical difference between the central intelligence indicators *CentrInt1* and *CentrInt2* is statistically significant and it is not the consequence of an accident/variability. From the classification point of view, *MAS1* and *MAS2* cannot be classified in the same intelligence class. If $CentrInt1 < CentrInt2$, then the conclusion is that *MAS1* is less intelligent than *MAS2*. If $CentrInt1 > CentrInt2$, then the conclusion is that *MAS1* is more intelligent than *MAS2*.

## 4. CASE STUDY FOR COMPARISON OF TWO MULTIAGENT SYSTEMS' INTELLIGENCE

For the validation of the proposed *MetrIntPair* metric, we designed a case study in order to prove its effectiveness. Here, we will investigate the comparison of two cooperative multiagent systems intelligence denoted *MAS1* and *MAS2*, formed of mobile agents. By mobile agents we understand simple computing agents able to move in the environment during the problem-solving. As the agents operate, the environment is constituted by a graph of connected nodes. An agent is able to move from one node to another. In each multiagent system, the agents cooperate in order to efficiently solve the undertaken problem by the multiagent system. The

communication of the agents is very simple by using signs. It is similar to the communication of the natural ants by using pheromones.

### 4.1. Presentation of the Studied Multiagent Systems

For validation purpose of the proposed metric, we realized experimental evaluations of more cooperative multiagent systems composed of simple cooperating agents applied for the *TSP* solving. We experimented: *Ant System*, *Elitist Ant System*, *Ranked Ant System*, *Best-Worst Ant System*, *Min-Max Ant System*, and *Ant Colony System*. Finally, we decided to present the validation for two cooperative multiagent systems whose central intelligence indicators as the numerical value is almost equal. We considered two multiagent systems composed of simple cooperating agents that mimic the behavior of natural ants. The ability of natural ants to determine the shortest paths between the nest and the food source was investigated in many studies.[57,58] Individual ant possesses few capabilities, but by operating in swarms/colonies, the ants are capable of a complex surviving behavior. Natural ants use chemical pheromones to communicate with each other, this allows the finding of the shortest trail between food sources and their nests. All of the ants that search for food deposit pheromone on the trail while walking. Each ant follows the pheromone trails that it meets with some probability, which is proportioned to the density of the pheromone. More ants walk on a trail, the more pheromone is deposited on it and more and more ants follow that trail. Based on this collaboration mechanism, it is a very high probability for the ants to find a short path. The obtaining of the shortest path, the optimal solution, is not guaranteed.

Marco Dorigo in the year 1992 in his Ph.D. thesis[59,60] firstly proposed that the problem-solving based on simple computing agents that mimic the operation of natural ants. Artificial ants (operate as reactive agents known in the intelligent agent literature) imitate the behavior of natural ants when they search for the food.

In an *Ant System*, initially, each agent (artificial ant) is placed on some randomly chosen city (node in the graph). An agent $k$ currently at a node $i$ choose to move to node $j$ by applying the following probabilistic transition rule (Equation 2). After each agent completes its tour, the pheromone amount on each path will be adjusted according to Equation 3.

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in J_k(i)} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} & \text{if } j \in J_k(i) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$\begin{aligned} \tau_{ij}(t+1) &= (1-\rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t) \\ \Delta\tau_{ij}(t) &= \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \\ \Delta\tau_{ij}^{k}(t) &= \begin{cases} \frac{Q}{L_k} & \text{if}(i,j) \in \text{ tour done by agent } k \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

In Equations 2 and 3, following notations are used: $\alpha$ and $\beta$ are adjustable parameters. $\alpha$ and $\beta$ control the relative weights of the heuristic visibility and the pheromone trail. In the parameters establishment, a trade-off between edge length and pheromone intensity appears to be necessary. $Q$ denotes an arbitrary constant. $d_{k,h}$ represents the distance between the nodes ($k$ and $h$); $\eta_{kh} = 1/d_{k,h}$ is the heuristic visibility of edge ($k$,$h$). $1 - \rho$ is the pheromone decay parameter, $0 < \rho < 1$, it represents the trail evaporation when the agent chooses a node to decide where to move. $m$ denotes the number of agents, $L_k$ denotes the length of the tour performed by the agent $k$.

In our experimental setup, *MAS1* operated as a *Rank-Based Ant System*[22,23] and *MAS2* operated as a *MMAS*.[23,24] *Rank-Based Ant System* and *MMAS* are well-known ant systems described in the scientific literature.

*Rank-Based Ant System* and *MMAS* have been applied for many problem-solving. Pérez-Delgado's paper[61] presents a solution for the Undirected Rural Postman Problem based on a *Rank-Based Ant System*. Thilagavathi and Amudha[62] present a solution for the 2D-HP Protein Folding using a *Rank-Based Ant Algorithm*. Stützle [63] proposes a solution for the quadratic assignment problem-solving using a *MMAS*.

We applied *MAS1* and *MAS2* for a class of NP-hard problem, *TSP* solving. *TSP* remains one of the most challenging problems in operational research.[17–20] *TSP* can be defined as follows: given $M$ cities, a salesman starts from a given node, should visit each node exactly one time and then returns home. He/she would like that the total distances (cost) traveled to be minimal (the smallest in one of the terms like: distance, time, money, or energy).

### 4.2. MAS1—The *Rank-Based Ant System*

In the *Rank-Based Ant System*,[22,23] the obtained solutions are ranked according to their length. The amount of deposited pheromone is then weighted for each solution. However, solutions with shorter paths deposit more pheromones than the solutions with longer paths. The difference from the traditional ant colony optimization algorithm is the pheromone update. For each of iterations, the best to date agent and additionally the $w - 1$ best agents for this iteration are selected. The best to date and each of the selected ranked agents deposit pheromone on the paths they travel:

$$\tau_{ij}(t+1) = \tau_{ij}(t) \cdot (1-\rho) + \sum_{r=1}^{r=w-1} e{\cdot}(w-r) \cdot \Delta\tau_{ij}^{r}(t) + e \cdot w \cdot \Delta\tau_{ij}^{bs}(t)$$
$$\Delta\tau_{ij}^{r}(t) = \frac{Q}{L^r} \tag{4}$$

In Equation 4, the following notations are used: $e$ is an additional multiplier, $e \geq 1$; $Q$ denotes a constant; $L^r$ denotes the length of the $r$th agent trip; $bs$ denotes the agent with the best to date trip; $\Delta\tau_{ij}{}^{bs}(t) = Q/L^{bs}$ if the path $ij \in T^{bs}$, $T^{bs}$ is the selected best to date agent's round trip; $L^{bs}$ is the length of the selected trip.

### 4.3. MAS2—Min-Max Ant System

*MMAS*[23,24] is an Ant Colony Optimization algorithm, a variant of the *Ant System*. *MMAS* differs from *Ant System* in some aspects. An *MMAS* gives dynamically evolving bounds on the pheromone trail intensities; this is made in such a way that the pheromone intensity on all the paths is always within a specified limit of the path with the greatest pheromone intensity. All the paths will have permanently a non-trivial probability of being selected. This way a wider exploration of the search space is assured. *MMAS* uses lower and upper pheromone bounds to ensure that all of the pheromone intensities are between the two bounds (lower and upper). In *MMAS*, the solution construction is according to Equation 2. There are variants in the selection of the agents that are allowed to update pheromones: the best for current iteration or best to date agent or the best after latest reset agent or the best to date agent for even (or odd) iterations. There are minimal and maximal pheromone limits to the quantity of pheromone on the paths between cities, denoted as $\tau_{\min}$ and $\tau_{\max.}$ The evaporation on the graph can be expressed as Equation 5. Equation 6 denotes the pheromone update based on the selected agent's round trip:

$$\tau_{ij}(t) = \max((1-\rho)\cdot\tau_{ij}(t), \tau_{\min}) \tag{5}$$

$$\tau_{ij}(t+1) = \min(\tau_{ij}(t) + \Delta\tau_{ij}^{bs}(t), \tau_{\max}) \tag{6}$$

where $\Delta\tau_{ij}^{bs}(t) = Q/L^{bs}$ if the path $ij \in T^{bs}$, $T^{bs}$ is the selected best to date agent's round trip. $L^{bs}$ is the length of the trip. It was used for initiation $\tau_0 = 1/$*number of cities*.

### 4.4. Experimental Results

In the experimental setup, we considered maps composed of 30 randomly placed cities ($nr = 30$). Based on some preliminary empirical evaluations, we established some parameter values as the most appropriate. The parameters of both multiagent systems *MAS1* and *MAS2* were considered: *number of steps* = 1200; $|MAS1| = 10$ the cardinality of *MAS1*, the number of agents of *MAS1*; $|MAS2| = 10$ the cardinality of *MAS2*. $\alpha = 1.738$, the power of the pheromone; $\beta = 2.085$, the power of the distance/edge weight; and *evaporation* = 0.163, the evaporation factor.

$|Prob| = 41$, where 41 represents the number of experimental intelligence evaluations for both *MASs*. This sample size, *Total sample size* = 41, can be obtained based on a calculus that compute the required sample size based on the given $\alpha$*_metr*, $\beta$*_metr*, *Power (1-*$\beta$*_metr), effect size*, and *number of tails* (one or two). We considered $\alpha\_metr = 0.05$, $\beta\_metr = 0.05$, $Power = 1 - \beta\_metr = 0.95$, *Effect size dz* = 0.58. We considered the case of the two-tailed test. Other outputs of the realized calculus are the *Noncentrality parameter* $\delta = 3.7138121$ and *Critical* $T = 2.0210754$.

Table III presents the obtained intelligence evaluation results for *MAS1* and *MAS2*. It was considered in both *MAS* as the intelligence indicator that obtained best to date travel value from the beginning of the problem-solving.

**Table III.** Results of the intelligence evaluation of *MAS1* and *MAS2*.

| *Rank-Based Ant System* (MAS1)/Set1 | *Min-Max Ant System* (MAS2)/Set2 |
|---|---|
| [*,#](Pr1)3.752; #(Pr2)5.442; (Pr3)6.106; [#,*](Pr4)4.513; (Pr5)7.084; (Pr6)6.012; (Pr7)6.931; (Pr8)6.068; (Pr9)6.548; (Pr10)6.076; (Pr11)6.345; (Pr12)6.73; (Pr13)6.595; (Pr14)7.028; (Pr15)6.393; (Pr16)7.263; (Pr17)6.28; (Pr18)7.096; (Pr19)6.212; (Pr20)6.373; (Pr21)7.672; (Pr22)7.888; (Pr23)6.259; (Pr24)6.606; (Pr25)6.39; (Pr26)6.579; (Pr27)6.828; (Pr28)6.932; (Pr29)6.364; (Pr30)6.488; (Pr31)6.553; (Pr32)6.632; (Pr33)7.61; (Pr34)6.922; (Pr35)7.215; (Pr36)6.935; (Pr37)7.196; (Pr38)5.75; (Pr39)6.78; (Pr40)7.122; (Pr41)6.926 | [#](Pr1)4.159; [#,**](Pr2)4.717; (Pr3)6.579; [#](Pr4)6.189; (Pr5)6.831; (Pr6)5.707; (Pr7)6.104; (Pr8)5.636; (Pr9)6.584; (Pr10)6.032; (Pr11)6.924; (Pr12)6.184; (Pr13)6.231; (Pr14)7.448; (Pr15)7.014; (Pr16)7.076; (Pr17)6.817; (Pr18)6.787; (Pr19)7.301; (Pr20)7.344; (Pr21)7.267; (Pr22)6.968; (Pr23)6.182; (Pr24)6.535; (Pr25)6.683; (Pr26)6.744; (Pr27)6.328; (Pr28)7.064; (Pr29)6.961; (Pr30)7.119; (Pr31)5.985; (Pr32)6.657; (Pr33)7.245; (Pr34)6.566; (Pr35)6.787; (Pr36)7.15; (Pr37)6.908; (Pr38)7.19; (Pr39)6.292; (Pr40)6.555; (Pr41)7.232 |

[*]Indicates an identified outlier in *Set1*.
[#]Indicates a value that is eliminated.
[**]Indicates an identified outlier in *Set2*.

In the case of the proposed metric, the selection of the intelligence indicator is the responsibility of the human evaluator who wishes to compare the problem-solving intelligence of the considered multiagent systems. He/she should choose it based on what could indicate the intelligence. An intelligence indicator value if necessary can be computed as a weighted sum of some other values that measure different aspects of a system's intelligence. In most of the studies, swarm systems that operate as ant colonies are considered intelligent. As human evaluators of the intelligence, we have considered the most representative global best as indicator of the intelligence.

Figure 1 contains the graphical representation of the intelligence indicators with all included measured intelligence data. The outlier intelligence values are not eliminated. Figure 1 illustrates the variability in intelligence of *MAS1* and *MAS2*.

Based on the *MetrIntPair* algorithm, as the first step indicated, it was realized a descriptive statistics[16,64] by computing the values for the Mean, Standard Error, Median, SD, Sample Variance, Min, Max, Lower CI, Upper CI, and *CV*. As *Confidence Level* of the mean in most of the cases, we propose the sellection of 95.0%; or expressed as significance level $\alpha = 0.05$. Table IV presents the results of the descriptive statistics realized for the *Set1* and *Set2*.

The data normality of *Set1* and *Set2* was verified with the Kolmogorov–Smirnov Goodness-of-Fit Test.[65] We chose to apply it at significance level $\alpha\_kolmogorov = 0.05$. The results of the normality test are presented in Table IV. The test results show that none of the intelligence indicator data (*Set1* and *Set2*) passed the normality test. The results indicate a moderate relatively homogeneous intelligence indicators data set.

Based on these obtained results, it was decided to search for outlier intelligence values. We chose to apply the Grubbs test for outliers' detection.[56] At the first step, we applied the Grubbs test for the *Set1* data. The test identified the

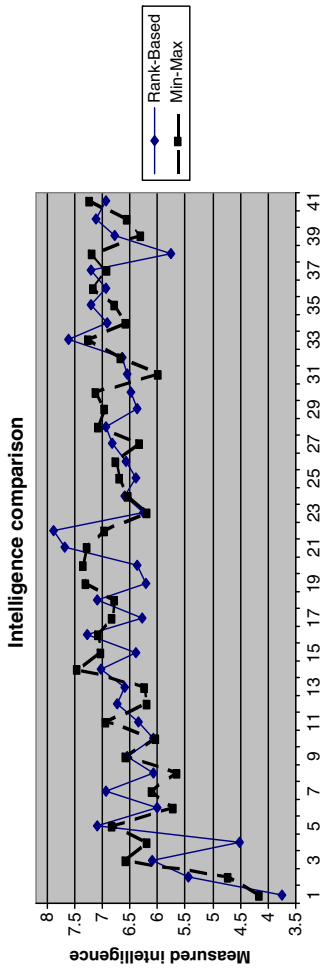



**Figure 1.** Graphical representation of intelligence indicators with the outlier intelligence indicator values included.

**Table IV.** Statistical analysis of the measured intelligence indicators.

| | *Rank-Based Ant System*/MAS1 | *Min-Max Ant System*/MAS2 |
|---|---|---|
| Mean/[Lower CI 95%,Upper CI 95%] | 6.5486341463/[6.310, 6.787] | 6.5873658537/[6.374, 6.8] |
| SD/Standard Error/Variance | 0.7549/0.1179/0.5699 | 0.675/0.1054/0.4456 |
| CV/Interpretation | 11.53/relatively homogeneous | 10.2/relatively homogeneous |
| Sample Size | 41 | 41 |
| Median/Minimum/Maximum | 6.595/3.752/7.888 | 6.744/4.159/7.448 |
| K-S Statistic/P-value/Passed | 0.141/0.0391/NO | 0.152/0.0181/NO |

corresponding intelligence value in *Set1* to *Pr1* was 3.752 (indicated with * in Table III), *Set1* = *Set1*-{3.752}. Based on the data pairing property, we decided also to eliminate the value 4.159 from the same position from *Set2*(corresponding to *Pr1*), *Set2* = *Set2*-{4.159}. We applied the test again for the new *Set1* and it identified a second outlier value in *Set1* corresponding to *Pr4*, *Set1* = *Set1*-{4.513}. We eliminated the corresponding value 6.189 from *Set2* to *Pr4*, *Set2* = *Set2*-{6.189}. After this step, no other outlier in the remaining *Set1* was identified. We applied the Grubbs test in the *Set1* for the third time but no other outlier was detected.

On the remaining data from *Set2*, we applied the Grubbs test. An intelligence indicator value as an outlier that corresponds to *Pr2*, 4.717, *Set2* = *Set2*-{4.717} was detected. Based on the data pairing, we eliminated the intelligence value 5.442 from *Set1* also, *Set1* = *Set1*-{5.442}. We applied the Grubbs test again for *Set2* but no other outlier was detected.

Figure 2 contains the graphical representation of intelligence indicators *Set1* and *Set2* obtained after the elimination of outlier intelligence values. Applying the outliers test, the sample size of both sets *Set1* and *Set2* was reduced to 38 (38 intelligence indicator measurements for both MASs). This sample size, *Total sample size* = 38, can be obtained based on a calculus that compute the required sample size based on the given $\alpha\_metr$, $\beta\_metr$, *Power* = *1 −0* $\beta\_metr$, *effect size,* and *number of tails* (one or two). We chose the following values: $\alpha\_metr = 0.05$, $\beta\_metr = 0.05$, *Power* = $1 - \beta\_metr = 0.95$, *Effect size dz* = 0.605. We considered the case of the two-tailed test. Other outputs of the realized calculus are the *Noncentrality parameter* $\delta = 3.7294705$ and *Critical T* = 2.0261925.

The obtained results presented in Table V shows that both intelligence data sets *Set1* and *Set2* obtained after the elimination of outliers passed the normality test. Further, the proposed *MetrIntPair* metric can be applied. The *Two Sample t-test* was applied for paired data.[66] We considered the significance level $\alpha\_metr = 0.05$. The obtained values were *P-value* = 0.9456, *T* = 0.06872, and *degrees of freedom* = 37. Based on the fact that *P-value* > $\alpha\_metr$, it can be decided that *MAS1* and *MAS2* intelligence is the same (there is no statistical difference between them).

We verified the assumption that the pairing was effective, according to the indication from the algorithm. This was verified by checking the existence of a linear correlation. Based on the normality of data sets *Set1* and *Set2* (data sampled from a Gaussian population), we considered the calculation of *Pearson Coefficient of Correlation*[67–69] denoted with *r* as the most appropriate. |*Set1*| = |*Set2*| = 38.

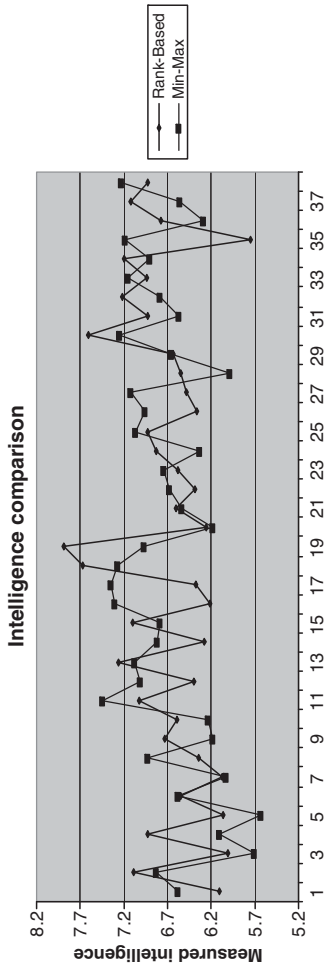



**Figure 2.** Graphical representation of intelligence indicators witouth the outlier intelligence indicator values included.

**Table V.** Statistical analysis on the measured intelligence indicators after the elimination of outlier intelligence indicators.

| | *Rank-Based Ant System*/MAS1 | *Min-Max Ant System*/MAS2 |
|---|---|---|
| Mean/[Lower CI 95%, Upper CI 95%] | 6.7049210526 [6.546, 6.864] | 6.7109736842 [6.557, 6.865] |
| SD/Standard Error/Variance | 0.4843/0.07857/0.2345 | 0.4676/0.07586/0.2186 |
| CV/Interpretation | 7.22/Homogeneouse data | 6.97/Homogeneous data |
| Sample Size | 38 | 38 |
| Median/Minimum/Maximum | 6.619/5.750/7.888 | 6.787/5.636/7.448 |
| Normality Test KS/*P* Value/Passed | 0.08615/>0.10/YES | 0.09089/>0.10/YES |

$r = 0.3498$ suggests the existence of positive correlation. As 95% CI for the correlation coefficient was obtained as [0.03384, 0.5022]. The obtained *Coefficient of Determination* was $r^2 = 0.1224$. The *Coefficient of Determination*[70] is a number that indicates the proportion of the variance in a variable (can be called dependent—in our case study is *Set2*) that is predictable from the other variable (can be called independent—in our case study is *Set1*).

We applied a statistical test for the verification if $r$ is statistically different from 0, verifying the hypothesis if a positive correlation exists. We have considered the significance level $\alpha\_cor = 0.05$. The result of the test was *P-value* $= 0.0313$, *P-value* $< \alpha\_cor$, which indicated a statistical difference.

## 5. DISCUSSIONS

We consider that it is impossible to give a general, unanimously accepted definition to the agent-based systems' (agents or cooperative multiagent systems) intelligence. Many definitions of agent's intelligence presented in the scientific literature are based on some biological considerations, such as autonomous learning, self-adaptation, or natural evolution realized during more generations. In a cooperative multiagent system, the intelligence can be considered at the system's level. Many studies proved that even very simple, efficiently and flexibly cooperating agents could form an intelligent multiagent system in that the intelligence emerges at the system's level.

There is not enough to formulate a general definition of cooperative multiagent system's intelligence supported only by some intuitive biologically inspired considerations. We consider that innovative metrics should be developed for an accurate measuring of a cooperative multiagent system's intelligence and for further comparisons with the intelligence of another multiagent system.

Our *MetrIntPair* metric is appropriate for multiagent systems, where the intelligence indicator for a problem-solving by the system can be expressed as a single value. This value can be computed as a weighted sum of some other values that measure different aspects of the system's intelligence.

The elaborated metric takes into consideration the variability in the intelligence of the compared multiagent systems. A multiagent system could have different intelligent reactions in different situations. In a specific situation, the reaction could

be more or less intelligent. In our research, we considered the outlier intelligence values, too high or too low intelligence values. They are much more different from the other intelligence values and if are taken into consideration, they could influence the measured MIQ.

For our metric, we considered important that the intelligence sample indicators data to be normally distributed. If the data is normally distributed then the mean should be chosen as a representative statistical indicator of the central intelligence tendency. The metric algorithm verifies the data normality. If the intelligence indicator data sets *Set1* and *Set2* are not normally distributed then a transformation is recommended (see Table V), or the opting for the elimination of outlier intelligence values.

In the paper [71] a metric called *MetrIntComp* for comparison of two cooperative multiagent system's intelligence is proposed. Both metrics *MetrIntComp* and *MetrIntPair* are based on the similar principle of difficult problem-solving intelligence measuring and classification in intelligence classes. The effectiveness of the metric *MetrIntComp* is proved by a case study. *MetrIntComp* is designed to be robust. This is based on the fact that in the *MetrIntComp* metric algorithm for the intelligence indicator data comparison uses the two unpaired-samples *Mann-Whitney* test that is known as a nonparametric robust test [72,73]. Mann-Whitney test for two unpaired-samples it is the non-parametric analog to the two-sample unpaired T-test.

*MetrIntPair* is more accurate based on the consideration that is based on normally distributed intelligence indicator data using by a parametric test. Parametric tests require data sets by a smaller size. In situations of non-normality, solutions like the elimination of outlier values or aplication of a transformation of the data are used. In the paper [74] is proved that for sample sizes larger than 30 ($>30$), the violation of the normality assumption should not cause major problems. In case of large enough sample sizes (hundreds), frequently can be used parametric methods even when the data are not normally distributed [75,76]. Another property of the *MetrIntPair* is the pairwise intelligence evaluations, and using the Paired Two Sample T-test, which has an increased accuracy for low numbers of intelligence indicator data.

## 6. CONCLUSIONS

Intelligent cooperative multiagent systems are used for many real-life difficult problems-solving. There are very few metrics that allows a quantitative comparison of two multiagent systems intelligence level/quotient. In this paper, we proposed a novel metric called *MetrIntPair* that allows an accurate comparison of two cooperative multiagent systems' intelligence in solving the same class(es)/type(s) of problems. The proposed metric takes into account the variability in the intelligence of the multiagent systems. By evaluating a multiagent system's intelligence in a specific situation, a result could occur; by repeating the evaluation, a different result could be obtained. For increasing the comparison accuracy for a smaller numbers of intelligence evaluations, the principle of intelligence indicators data pairing was

considered.

For validation purposes of the proposed metric, we designed a case study for two cooperative multiagent systems denoted as *MAS1* and *MAS2*. *MAS1* operated as a *Rank-Based Ant System*[22,23] and *MAS2* operated as an *MMAS*,[23,24] both of them being specialized in solving an NP-hard problem, namely the *TSP*. The result of the intelligence comparison based on the proposed metric proved that the obtained small difference in intelligence was the result of an accident/variability. By repeating the experiments, slightly different results could occur, but the conclusion related with the classification is the same, there is no statistically significant difference in the intelligence between the two considered multiagent systems.

Based on a comprehensive study of the scientific literature, we consider that our proposed metric called *MetrIntPair* is original, and will represent the basis for intelligence comparison of cooperative multiagent systems in many further researches. As an important property that makes it applicable is its universality. It could be applied in the comparison of any agent-based systems, agents that solve solely problems, or multiagent systems that cooperatively solve problems. It is not limited on aspects such as the agents' architecture, the multiagent systems' architecture. Using this metric, the intelligence of a multiagent system composed of reactive agents and the intelligence of a multiagent system composed of agents can be compared with logic architecture in solving problems by the same type.

## Acknowledgments

Laszlo Barna Iantovics acknowledge the support of the COROFLOW project PN-III-P2-2.1-BG-2016-0343.

## References

1. Liu S, Li T, Xie L. Distributed consensus for multiagent systems with communication delays and limited data rate. SIAM J Control Optim 2011;49(6):2239–2262.
2. Neri F. Agent-based modeling under partial and full knowledge learning settings to simulate financial markets. AI Commun 2012;25(4):295–304.
3. Addis V, Armano G, Vargiu E. Multiagent systems and information retrieval our experience with X.MAS. Expert Syst Appl 2012;39(3):2509–2523.
4. Cena CG, Cardenas PF, Pazmino RS, Puglisi L, Santonja RA. A cooperative multi-agent robotics system: Design and modeling. Expert Syst Appl 2013;40(12):4737–4748.
5. Stoean C, Stoean R. Cooperative coevolution, support vector machines and evolutionary algorithms for classification, intelligent systems reference library. Berlin, Germany:Springer; 2014;69:57–73.
6. Geetha DD, Nalini N, Biradar RC. Agent based trusted neighbor identification in wireless sensor networks. AI Commun 2015;28(4):693–708.
7. Baykasoğlu A, Kaplanoğlu V. An application oriented multi-agent based approach to dynamic load/truck planning. Expert Syst Appl 2015;42(15–16):6008–6025.
8. Schreiner K. Measuring IS: Toward a US standard. IEEE Intell Syst App 2000;15(5):19–21.
9. Kannan B, Parker LE. Metrics for quantifying system performance in intelligent, fault-tolerant multi-robot teams. 2007 IEEE/RSJ Int Conf Intell Robots Syst San Diego, CA, United States of America; 2007;951–958.

10. Anthon A, Jannett TC. Measuring machine intelligence of an agent-based distributed sensor network system. In: Elleithy K, editor. Advances and innovations in systems, computing sciences and software engineering. Netherlands:Springer; 2007. pp 531–535.
11. Hibbard B. Measuring agent intelligence via hierarchies of environments. In: Schmidhuber J, Thórisson KR, Looks M, editors. Artificial general intelligence. Lect Notes Comput Sci 2011;6830:303–308.
12. Wallin JEW. The new clinical psychology and the psycho-clinicist. J Educ Psychol 1911;2(3):121–132.
13. Terman LM, Lyman G, Ordahl G, Ordahl L, Galbreath, N, Talbert W. The stanford revision of the Binet-Simon scale and some results from its application to 1000 non-selected children. J Educ Psychol 1915;6(9):551–562.
14. Neisser U. Rising scores on intelligence tests. Am Sci 1997;85:440–447.
15. Richardson JTE. Howard Andrew Knox and the origins of performance testing on Ellis Island, 1912–1916. Hist Psychol 2003;6(2):143–170.
16. Nick TG. Descriptive statistics. Topics in biostatistics. Method Mol Biol 2007;404:33–52.
17. Rosenkrantz DJ, Stearns RE, Lewis II PM. An analysis of several heuristics for the traveling salesman problem. SIAM J Comput 1977;6(3):563–581.
18. Hartvigsen D, Pulleyblank WR. Outer-facial graphs and the traveling salesman problem. SIAM J Optim 1994;4(3):676–689.
19. Burkard RE, Deineko VG, Dal RV, Veen JAAVD, Woeginger GJ. Well-solvable special cases of the traveling salesman problem: A survey. SIAM Rev 1998;40(3):496–546.
20. van Leeuwen Jan, Ed. Handbook of theoretical computer science. Vol. A, Algorithms and complexity. Amsterdam, The Netherlands: Elsevier; 1998.
21. Crisan GC, Nechita E, Palade V. Ant-based system analysis on the traveling salesman problem under real-world settings. In: Hatzilygeroudis I, Palade V, Prentzas J, editors. Combinations of intelligent methods and applications. Smart Innovation, Systems and Technologies. Switzerland: Springer; 2016. 46: pp 39–59.
22. Bullnheimer B, Hartl RF, Strauss C. A new rank based version of the ant system. A computational study. Cent Eur J Oper Res 1999;7(1):25–38.
23. Prakasam A, Savarimuthu N. Metaheuristic algorithms and probabilistic behaviour: A comprehensive analysis of ant colony optimization and its variants. Artif Intell Rev 2016;45(1):97–130.
24. Stützle T, Hoos HH. MAX MIN ant system. Fut Gen Comp Syst 2000;16(8):889–914.
25. Iantovics LB. A new intelligent mobile multiagent system. Proc. IEEE-SOFA 2005, Szeged-Hungary and Arad-Romania: IEEE CS Press; 2005. pp 153–159.
26. Langley P, Laird JE, Rogers S. Cognitive architectures: Research issues and challenges. Cognitive Sys Res 2009;10(2):141–160.
27. Iantovics LB, Zamfirescu CB. ERMS: An evolutionary reorganizing multiagent system. Innov Comput Inform Control 2013;9(3):1171–1188.
28. Dastani M, Meyer JJCh. Agents with emotions. Int J Intell Sys 2010;25(7):636–654.
29. Irani S, Rabani Y. On the value of coordination in distributed decision making. SIAM J Comput 1996;25(3):498–519.
30. Yang K, Galis A, Guo X, Liu D. Rule-driven mobile intelligent agents for real-time configuration of IP networks, knowledge-based intelligent information and engineering systems. Lecture Notes Comput Sci 2003;2773:921–928.
31. West D, Dellana S. Diversity of ability and cognitive style for group decision processes. Inform Sci 2009;179(5):542–558.
32. Zamfirescu CB, Duta L, Iantovics LB. On investigating the cognitive complexity of designing the group decision process. Stud Inform Control 2010;19(3):263–270.
33. Guillaud A, Troadec H, Benzinou A, Bihan JL, Rodin V. A multiagent system for edge detection and continuity perception on fish otolith images. EURASIP J Appl Signal Process 2002;7:746–753.
34. Jumadinova J, Dasgupta P. A multi-agent system for analyzing the effect of information on prediction markets. Int J Intell Sys 2011;26(5):383–409.

35. Fox J, Beveridge M, Glasspool D. Understanding intelligent agents: Analysis and synthesis. AI Commun 2003;16(3):139–152.
36. Chmait N, Dowe DL, Green DG, Li YF. Observation, communication and intelligence in agent-based systems. In: Bieger J, Goertzel B, Potapov A, editors. Artificial general intelligence. Lecture Notes Comput Sci. 2015;9205:50–59.
37. Chmait N, Dowe DL, Green DG, Li YF, Insa-Cabrera J. Measuring universal intelligence in agent-based systems using the anytime intelligence test. Technical Report, Monash University; Report Number 2015/279; 2015.
38. Legg S, Hutter, M. A formal measure of machine intelligence. In: 15th Annu Mach Learn Conf Belgium and The Netherlands, Benelearn 2006, Ghent; 2006. pp 73–80.
39. Legg S, Hutter M. Universal intelligence: A definition of machine intelligence. J Minds Mach Arch 2007;17(4):391–444.
40. Bell EA, Boehnke P, Harrison TM, Mao, WL. Potentially biogenic carbon preserved in a 4.1 billion year-old zircon. Proc Natl Acad Sci USA 2015;112(47):14518–14521.
41. Hibbard B. Bias and no free lunch in formal measures of intelligence. J Artif Gen Intell 2009;1(1):54–61.
42. Legg S, Veness J. An approximation of the universal intelligence measure. In: Dowe DL, editor. Algorithmic probability and friends. Bayesian prediction and artificial intelligence. Lecture Notes Comput Sci 2013;7070:236–249.
43. Solomonoff RJ. A formal theory of inductive inference. Part I. Inform Control 1964;7(1):1–22.
44. Solomonoff RJ. A formal theory of inductive inference. Part II. Inform Control 1964;7(2):224–254.
45. Solomonoff RJ. Complexity-based induction systems: comparisons and convergence theorems. IEEE Trans Inform Theory 1978;24:422–432.
46. Winklerová Z. Maturity of the particle swarm as a metric for measuring the collective intelligence of the swarm. In: Tan Y, Shi Y, Mo H, editors. Advances in swarm intelligence. Lecture Notes Comput Sci 2013;7928:40–54.
47. Hernández-Orallo J, Dowe DL. Measuring universal intelligence: Towards an anytime intelligence test. Artif Intell 2010;174(18):1508–1539.
48. Chmait N, Li YF, Dowe DL, Green DG. A dynamic intelligence test framework for evaluating ai agents. In: Proc Workshop Evaluating General-Purpose AI; EGPAI 2016, 2016;1–8.
49. Motulsky H. GraphPad InStat Version 3. The InStat guide to choosing and interpreting statistical tests. San Diego, CA, United States of America:GraphPad Software, Inc. 2003.
50. Everitt B. The cambridge dictionary of statistics. Cambridge, New York: Cambridge University Press; 1998.
51. Marusteri M, Bacarea V. Comparing groups for statistical differences: how to choose the right statistical test? Biochem Med 2010;20(1):15–32.
52. Ross SM. Peirce's criterion for the elimination of suspect experimental data. J Eng Technol 2003;20(2):38–41.
53. Zerbet A, Nikulin M. A new statistics for detecting outliers in exponential case. Commun Stat: Theor Method 2003;32(3):573–583.
54. Stigler SM. Mathematical statistics in the early states. Ann Stat 1978;6:239–275.
55. Dean RB, Dixon WJ. Simplified statistics for small numbers of observations. Anal Chem 1951;23(4):636–638.
56. Barnett V, Lewis T. Outliers in statistical data, 3rd ed. New York, United States of America: John Wiley & Sons; 1994.
57. Higashi S, Yamauchi K. Influence of a supercolonial Ant Formica (Formica) yessensis Forel on the distribution of other ants in Ishikari coast. Jpn J Ecol 1979;29:257–264.
58. Giraud T, Pedersen JS, Kelle L. Evolution of supercolonies: The Argentine ants of southern Europe. Proc Natl Acad Sci USA 2002;99(9):6075–6079.
59. Colorni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies, Actes de la première conférence européenne sur la vie artificielle. Paris, France: Elsevier Publishing; 1991. pp 134–142.

60. Dorigo M. Optimization, learning and natural algorithms, PhD Thesis, Politecnico di Milano, Italy; 1992.
61. Pérez-Delgado ML. Rank-based ant system to solve the undirected rural postman problem. In: Omatu S, Rocha MP, Bravo J, Fernández F, Corchado E, Bustillo A, Corchado JM, editors. Distributed computing, artificial intelligence, bioinformatics, soft computing, and ambient assisted living. Lecture Notes Comput Sci 2009;5518:507–514.
62. Thilagavathi N, Amudha T. Rank based ant algorithm for 2D-HP protein folding. In: Jain L, Behera H, Mandal J, Mohapatra D, editors. Computational intelligence in data mining - Vol. 3, smart innovation, systems and technologies - Vol. 33. New Delhi: Springer; 2015. pp 441–451.
63. Stützle T. MAX-MIN ant system for the quadratic assignment problem, Technical Report AIDA-97–4, TU Darmstadt, Germany: FB Informatik; 1997.
64. Mann PS. Introductory statistics, 2nd ed. New York, United States of America:Wiley; 1995.
65. Chakravarti IM, Laha RG, Roy J. Handbook of methods of applied statistics, Vol I. New York, United States of America:John Wiley & Sons; 1967;392–394.
66. Lowry R. Concepts & Applications of Inferential Statistics. http://vassarstats.net/textbook
67. Galton F. Regression towards mediocrity in hereditary stature. J Anthropol Inst Great Britain Ireland 1886;15:246–263.
68. Pearson K. Notes on regression and inheritance in the case of two parents. P Roy Soc Lond 1895;58:240–242.
69. Stigler SM. Francis Galton's account of the invention of correlation. Stat Sci 1989;4(2):73–79.
70. Draper NR, Smith H. Applied regression analysis. New York, United States of America: Wiley-Interscience; 1998.
71. Iantovics LB, Rotar C, Nechita E. A novel robust metric for comparing the intelligence of two cooperative multiagent systems. Proc. 20th Int Conf Knowl Based Intell Inform Eng Syst, KES2016, September 5–7, 2016, York, United Kingdom, Procedia Computer Science; 2016:96. pp 637–644.
72. Mann HB, Whitney DR. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. Annals of Mathematical Statistics 1947;18(1):50–60.
73. Fay MP, Proschan MA. Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. Statistics Surveys 2010;4:1–39.
74. Pallant J. SPSS survival manual, a step by step guide to data analysis using SPSS for windows, 3 ed., Sydney. Australia: McGraw Hill; 2007. pp 179–200.
75. Elliott AC, Woodward WA. Statistical analysis quick reference guidebook with SPSS examples, 1st ed., London: Sage Publications; 2007.
76. Altman DG, Bland JM. Statistics notes: the normal distribution. BMJ 1995;310(6975):298.

symmetry

MDPI

Article

# MetrIntSimil—An Accurate and Robust Metric for Comparison of Similarity in Intelligence of Any Number of Cooperative Multiagent Systems

**Laszlo Barna Iantovics [1,*,†], Matthias Dehmer [2,3,4,†] and Frank Emmert-Streib [5,†]**

1 Department Informatics, Petru Maior University, 540088 Targu Mures, Romania
2 Department Production and Operations Management, Faculty of Management, University of Applied Sciences Upper Austria, Campus Steyr, Wehrgrabengasse 1-3, 4400 Steyr, Austria; matthias.dehmer@umit.at
3 College of Computer and Control Engineering, Nankai University, 300071 Tianjin, China
4 UMIT, Department of Mechatronics and Biomedical Computer Science, Eduard Wallnoefer Zentrum 1, 6060 Hall, Austria
5 Computational Medicine and Statistical Learning Laboratory, Department of Signal Processing, Tampere University of Technology, 33720 Tampere, Finland; frank.emmert-streib@tut.fi
* Correspondence: ibarna@science.upm.ro; Tel.: +40-265-262275
† These authors contributed equally to this work.

Received: 26 December 2017; Accepted: 9 February 2018; Published: 14 February 2018

**Abstract:** Intelligent cooperative multiagent systems are applied for solving a large range of real-life problems, including in domains like biology and healthcare. There are very few metrics able to make an effective measure of the machine intelligence quotient. The most important drawbacks of the designed metrics presented in the scientific literature consist in the limitation in universality, accuracy, and robustness. In this paper, we propose a novel universal metric called *MetrIntSimil* capable of making an accurate and robust symmetric comparison of the similarity in intelligence of any number of cooperative multiagent systems specialized in difficult problem solving. The universality is an important necessary property based on the large variety of designed intelligent systems. *MetrIntSimil* makes a comparison by taking into consideration the variability in intelligence in the problem solving of the compared cooperative multiagent systems. It allows a classification of the cooperative multiagent systems based on their similarity in intelligence. A cooperative multiagent system has variability in the problem solving intelligence, and it can manifest lower or higher intelligence in different problem solving tasks. More cooperative multiagent systems with similar intelligence can be included in the same class. For the evaluation of the proposed metric, we conducted a case study for more intelligent cooperative multiagent systems composed of simple computing agents applied for solving the *Symmetric Travelling Salesman Problem (STSP)* that is a class of NP-hard problems. *STSP* is the problem of finding the shortest Hamiltonian cycle/tour in a weighted undirected graph that does not have loops or multiple edges. The distance between two cities is the same in each opposite direction. Two classes of similar intelligence denoted *IntClassA* and *IntClassB* were identified. The experimental results show that the agent belonging to *IntClassA* intelligence class is less intelligent than the agents that belong to the *IntClassB* intelligence class.

**Keywords:** symmetric travelling salesman problem; diversity of intelligent systems; similarity in intelligence; machine intelligence measure; cooperative problem solving; computational-hard problem

## 1. Introduction

Intelligent cooperative multiagent systems *(ICMASs)* are applied for a large diversity of difficult real-life problem solving tasks [1–6]. In cooperative multiagent systems *(CMASs)*, the intelligence

could be considered at the system's level. Much research has [3,7] proved that, even in cooperative multiagent systems composed of simple agents, an increased intelligence at the systems level could emerge if the member agents cooperate efficiently and flexibly.

In scientific literature, the intelligence estimation is many times based on some intuitive biologically inspired intelligence considerations. There are very few studies related to intelligence measurement, from which few are able also to be applied for accurate and robust symmetric comparison of the intelligence of two or even more multiagent systems. We consider that, similarly to biological systems, there is a variability even in the case of *ICMASs*. In some situations, a *CMAS* could behave more intelligently, in others less intelligently. Another aspect consists in treating what we call low and high *outlier intelligence values*. Sometimes taking into consideration such values could result in an erroneous evaluation and/or comparison result. Another aspect that we consider important consists in the number of compared multiagent systems. If the intelligence of more than two multiagent systems should be compared, an erroneous decision consists in comparing them pairwise. We will further elaborate on this subject in the discussions section.

We have not found in the scientific literature an effective metric that includes all of the considerations previously mentioned, able to simultaneously compare the similarity in intelligence of any number (even more than two) cooperative multiagent systems. With this purpose, we propose a novel metric called *MetrIntSimil* that is capable of making an accurate and also robust symmetric comparison of the similarity in intelligence of two or more than two cooperative multiagent systems, taking into consideration the variability in the intelligence of the multiagent systems. The *Travelling Salesman Problem* (*TSP*) [8–10] is a well known NP-hard problem (non-deterministic polynomial-time hardness). NP-hardness, is the defining property of a class of problems that are, "at least as hard as the hardest problems in NP". Both variants of the *TSP* the *Symmetric TSP* (*STSP*) and the *Asymmetric TSP* (*ATSP*) are frequently studied. In order to assess the effectiveness of the proposed metric, we conducted a case study. Three multiagent systems composed of cooperative reactive agents specialized in solving a class of NP-hard problems, the *STSP* [11], were considered. The cooperative multiagent systems operated as a *Best-Worst Ant System* [12,13], a *Min-Max Ant System* [14,15] and an *Ant Colony System* [16,17].

The upcoming part of the paper is organized as follows: Section 2 analyzes the intelligence of cooperative multiagent systems; representative metrics described in the scientific literature proposed for measuring artificial systems intelligence are also presented. In Section 3, our proposed *MetrIntSimil* metric for intelligence comparison of more cooperative multiagent systems is presented. For validation purposes, in Section 4, a case study was performed. In Section 5, we discussed on the designed *MetrIntSimil* metric and compared it with a recent metric presented in the literature. Section 6 presents our next research direction. In Section 7, the main conclusions of the research described in this paper are presented.

## 2. Measuring the Machine Intelligence Quotient

### *2.1. Intelligent Cooperative Multiagent Systems*

Theories and principles of systems science can explain many complicated matters of the world and offer a new vision on many unsettled problems [18,19]. As a particular scientific domain, it has its own particular methodology for qualitative and quantitative analyses. The motivation for the domain of systems science is the necessity of understanding the systems research with operable mathematical methods as an organic whole. Systems science among others establishes systemic formulas based on a unified systems approach.

Langley, Laird, and Rogers [20] examine the motivations for research on cognitive architectures. In the paper, the architecture of cognitive systems described in scientific literature was reviewed. The authors discuss some open issues that should drive research related to the architecture of cognitive systems. They consider the following capabilities that a cognitive architecture should

support: organization, performance, learning and some theoretical criteria for making an evaluation at the system level.

In Ref. [21], a prototype agent-based geographical information system (GIS) abbreviated as *GXGIS* is proposed. In the proposal, a developed interface agent interactively assists the users in the query formation processes. It has the capability to offer domain knowledge associated with a given query. Another capability of the agent consists in the recording of a user's troubleshooting experience and show it to the same user or other users as hints.

We call intelligent agent-based systems the intelligent agents and the intelligent cooperative multiagent systems. The motivation for using this generic name consists in the fact that a cooperative multiagent system from the external point of view could be seen as an individual. Whomever (human or an artificial agent) submits a problem for solving does not know that the problem is solved by an agent or cooperatively by more agents. He/she/it submits the problem to an agent and the problem could be subsequently solved, if necessary, cooperatively by more agents. There is various research worldwide focused on the development of intelligent agent-based systems [2,3,22,23].

The agents' intelligence could not be unanimously defined based on the large diversity of agents [2]. The intelligence estimation is many times realized based on different considerations like [3]: autonomous learning, self-adaptation and evolution. These considerations are inspired by biological systems able to learn during their life cycle, to adapt to the environment and to evolve during more generations.

Mobile agents versus static agents are able to move in the environment during the problem solving. Mobile agents could be classified as software mobile agents and robotic mobile agents. The software mobile agents operate in a software environment (a computer, or a computer network); the mobile robotic agents operate in a physical environment (a swarm of mobile robots distributed in a physical environment specialized in collecting objects, for example).

To illustrate the impossibility of the definition of the agents' intelligence, let us consider the differences in intelligence between static software agents vs. mobile software agents [2]. Many developed mobile agents are more limited in intelligence than their static counterparts. Limitations in the mobile agents' endowment with intelligence are based on some practical reasons. The endowment of a mobile agent with intelligence may increase the agent's body size. The execution of an intelligent mobile agent (more complex software code, more computations) in a software environment requires more computational resources. The transmission of a large number of intelligent mobile agents in a network might overload the network with data transmission. A large number of intelligent mobile agents, which execute complex computations to a host, may overload that host. The mobile agents migrate during their operation in the network, and, based on this fact, it is difficult to estimate where a mobile agent is at a specific moment of time. This limitation makes the endowment of mobile agents with communication capacity difficult. The communication is indispensable for cooperation.

There is no unanimously accepted definition of the cooperative multiagent systems intelligence. This fact is based on the large diversity of cooperative multiagent systems. Many times, in scientific literature, a cooperative multiagent system is considered intelligent, based on the simple consideration that the efficient and flexible cooperation between the agents emerge in intelligence at the systems level. Based on this aspect, in a cooperative multiagent system, the intelligence could be considered at the system's level [2,3]. The intelligence in such a system is higher than the individual member agents' intelligence. Even in a very simple cooperative multiagent system, intelligence at the system's level could emerge. Efficient and flexible cooperating simple agents could intelligently solve difficult problems.

There is a lot of research [24,25] focused on the study of decision-making in the frame of cooperative coalitions. Decisions made in the frame of coalitions outperform many times the decisions of individuals that operate in isolation.

Yang, Galis, Guo, and Liu [7] present an intelligent mobile multiagent system composed of simple reactive agents. The mobile agents are specialized in a computer network administration. They are endowed with knowledge retained as a set of rules that describe network administration tasks. The multiagent system could be considered intelligent based on the fact that it simulates the behavior of a human network administrator.

In our approach, we will refer to agent-based systems, cooperative multiagent systems composed of two or more agents that cooperatively solve difficult problems. The agent members of such a system are not necessarily intelligent, but, at the level of the system, there is emerging increased intelligence that can be quantitatively measured. The study of aspects related to cooperative multiagent system intelligence is important in order to develop highly efficient problem solving methods.

### *2.2. Metrics for Measuring the Machine Intelligence*

The existence of some properties of the cooperative multiagent systems that could be associated with intelligence does not allow a quantitative evaluation, such properties just prove its existence. In Ref. [26], it is considered that the evaluation of a system's intelligence must be based on some effective metrics that allow the measuring of the quantity of intelligence and comparison of a system's intelligence with the intelligence of another system. In many papers, some evaluation or analysis of the system's intelligence are presented. There are very few designed metrics able to make an effective comparison of two or more multiagent systems' intelligence.

Besold, Hernandez-Orallo, and Schmid [27] study the difficult problems for the humans that could be used as benchmark problems for intelligent systems. The authors consider that an intelligent system that can successfully solve difficult problems for humans has some kind of human-level intelligence. The paper analyses and discusses this assumption.

There are some metrics developed for making different kinds of measurements in systems that are considered intelligent. Such metrics are not always developed for measuring the systems' intelligence as a whole. They are used only for measuring some aspects that represent interest. A fault-tolerant system should be able to diagnose and recover from some faults. The fault-tolerance has a direct impact on the performance of a system. Sometimes, this property of the systems is associated with the intelligence. Kannan and Parker [28] analyze the intelligent fault-tolerant systems. They study the effectiveness of some metrics for the evaluation of fault-tolerance in the context of system performance. A main focus of the presented approach is to capture the effect of intelligence (reasoning and learning) on the effective fault-tolerance of a system. The fault-tolerance of different heterogeneous multi-robot teams there were analyzed and compared. The proposed metric is able to make some evaluation of the quality of learning towards system level fault-tolerance.

Park, Kim, and Lim [29] analyzed the measuring of machine intelligence of human–machine cooperative systems. They proposed a so-called intelligence task graph as a modeling and analysis tool. The authors consider that a human–machine cooperative system can be modelled as equations.

Schreiner [30] presents a study realized by the US National Institute of Standards and Technology (NIST). It accentuates the necessity of developing intelligent systems in the US in different fields such as the industry and military, including public and private sectors. The study is related to creating standard measures for systems that can be considered intelligent. Schreiner accentuates the main studied questions related to how precisely intelligent systems are defined and how to measure and compare the capabilities that intelligent systems should provide. NIST's initial approach to establishing metrics attempts to address different pragmatic and theoretical aspects.

The paper [31] proposes a method called *OutIntSys* for the detection of the systems, which has a statistically extremely low or extremely high intelligence, from a set of intelligent systems that solves the same type(s) of problems. The proposed method can be applied in choosing the most intelligent systems from a set of intelligent systems able to solve difficult problems.

The Minimum Message Length (*MML*) principle supports a specific compression as a method to perform inductive inference resulting in intelligence [32–34]. Dowe and Hajek [35,36] proposed

an adapted Turing test with some specific compression exercises, having the purpose to measure the ability of inductive inference in the context of *MML* [37]. Sterret [38] analyzed how IBM developed a question-answering computer (Watson) competes against humans on the Jeopardy game. Watson [39] was developed in the frame of the DeepQA project by a research team led by David Ferrucci at "Thomas J. Watson Research Center" located in Yorktown Heights, New York, U.S.

Legg and Hutter defined a formal measure [40], presuming that the performance in easy environments counts more toward an agent's intelligence than does performance in difficult environments. Hibbard [41] proposed an alternative measure, which is based on a hierarchy of sets of increasingly difficult environments, considering a reinforcement learning framework. Hibbard considers an agent's intelligence as the ordinal of the most difficult set of environments that it can pass. The applied measure is defined in Turing machine and finite state machine models of computing. In the finite state machine model, the measure is calculated as the number of time steps necessary to pass the test.

Anthon and Jannett [42] define the agent-based systems intelligence based on the ability to compare alternatives with different complexity. In the presented approach, a measure of machine intelligence allows the comparing of alternatives with different complexity. A method for measuring the *Machine Intelligence Quotient* (*MIQ*) of the human-cooperative system is adapted and applied to measure the *MIQ* of an agent-based system. The method is proposed to be applied for agent-based distributed sensor network systems. The proposal was tested by comparing the *MIQ* in different agent-based scenarios for a distributed sensor network application.

Hernandez-Orallo and Dowe [43] proposed the idea of a general test called a universal anytime intelligence test. The authors of the study consider that such a test should be able to measure the intelligence level, which could be in different situations very low (called inept systems) or very high (called brilliant systems), of any biological or artificial system. Such a test should be able to evaluate slowly and quickly operating systems. Another property of such a test consists in the fact that it could be interrupted at any time, this way obtaining an approximation of the intelligence score. Based on this property, if more time is left for a test, then a more accurate result will be obtained. The proposed test is based on some previous works on the measurement of machine intelligence. The considered works were based on Kolmogorov complexity and universal distributions. In the 1990s, the C-tests and compression-enhanced Turing tests were developed. The proposal presented in the paper is also based on the idea of measuring intelligence through dynamic tests. The authors of the research discuss different developed tests by highlighting their limitations. They introduce some novel ideas that they consider necessary for the development of a "universal intelligence test".

A novel metric called *MetrIntMeas* for measuring the machine intelligence of a swarm system is presented in the paper [44]. *MetrIntMeas* is able to measure the machine intelligence of an evaluated swarm system and compare it with a considered reference machine intelligence value. The metric also makes a classification of the studied swarm system, by verifying if it belongs to the class of swarm systems with the considered reference machine intelligence value. The paper gives a definition to the swarm systems' evolution in intelligence. It defines the evolution of the swarm systems in the intelligence, as a measurable increase in intelligence by using the *MetrIntMeas* metric.

Many researchers are focused on the study of collective intelligence of the swarm systems. Many difficult problem solving tasks are based on swarm systems. Winklerova [45] assessed the collective intelligence of a particle swarm system according to a proposed Maturity Model. The proposed model is based on the Maturity Model of C2 (Command and Control) operational space and the model of Collaborating Software. The main aim of the study [45] was to obtain a more thorough explanation of how the intelligent behavior of the particle swarm emerges. A conclusion of the research is that a swarm system's effectiveness can be improved by adaptation of the rules that specifies the particle's behaviour. Each particle should adjust its velocity using some control parameters. The parameters value would be derived from inside of the swarm system.

Ref. [46] proposes a novel metric called *MetrIntPair* for measuring the machine intelligence of cooperative multiagent systems. The *MetrIntPair* metric is able to exactly analyze the intelligence of

two cooperative multiagent systems. It makes an accurate comparison of the intelligence of the two studied cooperative intelligent systems at an application, and at the same time it verifies if they can be included in the same class of intelligence (solve problems with the same intelligence). The intelligence comparison of two cooperative systems is based on some kind of pairwise problem solving intelligence evaluations. This approach has as principal advantage versus the non-pairwise intelligence evaluations the decrease of the number of necessary problem solving experimental intelligence evaluations versus the case when non-pairwise intelligence evaluations are made.

Liu, Shi, and Liu [47] present a study related to the analysis of the intelligence quotient and the grade of artificial intelligence. A so-called "standard intelligence model" was proposed. It unifies artificial intelligence and "human characteristics" that include aspects of knowledge like: input, output, mastery, and creation.

Douglas Detterman [48] announced a challenge related to the Artificial Intelligence artefact measuring by classical IQ tests. Sanghi and Dowe [49] in 2003 presented a computer program tested on some standard human IQ tests. The program was smaller than the chess-playing Deep Blue [50], having just about 960 lines of code written in the Perl programming language. It surpassed the average human intelligence score by 100 on some tests [49].

In this subsection, we presented some relevant metrics for measuring machine intelligence. Each of the proposed metrics is based on a specific method of measuring the intelligence, appreciating the intelligence measure based on some specific principles/definition of the intelligence. Different definitions of the intelligence and different associated measuring approaches do not allow a punctual comparison of the metrics (many of them do not allow even any considerable comparison). There is no standardization or even a relatively generally acceptable view on what an intelligence metric should measure. Based on this consideration, the design of novel universal metrics that could standardize the intelligence measuring is an open and important research direction.

In our opinion, the most feasible consideration of measuring the machine intelligence consists in the principle of difficult problem solving ability. The purpose of the design of intelligent systems consists in the efficient solving of difficult problems. We consider that the main purpose of an intelligence metric consists in the differentiation of the computing systems based on the problem solving intelligence. The universal *MetrIntComp* metric [51] presented in the literature is able to make a robust comparison of two cooperative multiagent system's intelligence, and classify them in intelligence classes. *MetrIntComp* is effective even in the case of small differences in intelligence between the compared intelligent systems. For proving the effectiveness of the metric, a case study for two cooperative multiagent systems was presented [51]. *MetrIntComp* has as disadvantages related to the reduced accuracy and the limitation in the application for intelligence comparison to more than two intelligent systems at the same time. We will discuss this subject with more details in the Discussion section. Based on these limitations, the design of a novel metric that eliminates these limitations is important and actual.

## 3. Description of the MetrIntSimil Metric

In this section, we present a novel universal metric proposed for the accurate and robust symmetric comparison of the similarity in intelligence of two or more than two *CMASs* specialized in difficult problem solving. The *MetrIntSimil* metric is described as an algorithm called from now on *Multiagent Systems Intelligence Comparison*. Henceforth, we consider a set of cooperative multiagent systems denoted as $MasSet = \{MA_1, MA_2, \ldots, MA_k\}$. $|MasSet| = k$ represents the number of compared multiagent systems. The obtained intelligence indicators as a result of problem solving intelligence measuring are denoted as $Set1 = \{A1t_1, A1t_2, \ldots, A1t_{n1}\}$, $Set2 = \{A2t_1, A2t_2, \ldots, A2t_{n2}\}, \ldots, Setk = \{Akt_1, Akt_2, \ldots, Akt_{nk}\}$. $|Set1| = n1, |Set2| = n2, \ldots, |Setk| = nk$ represents the cardinality/sample size of $Set1$, $Set2$, $\ldots$, $Setk$. Table 1 presents the obtained intelligence indicator results for $MasSet$. $A1t_1, A1t_2, \ldots, A1t_{n1}$—represents the measured intelligence of the $MA_1$; $A2t_1, A2t_2, \ldots,$

$A2t_{n2}$—represents the measured intelligence of the $MA_2$; ...; $Akt_1, Akt_2, \ldots, Akt_{nk}$—represents the measured intelligences of the $MA_k$.

**Table 1.** Symmetric experimental evaluation of $MA_1(Set_1)$, $MA_2(Set_2)$ and $MA_k(Set_k)$ intelligence.

| $Set_1$ | $Set_2$ | ... | $Set_k$ |
|---|---|---|---|
| $A1t_1$ | $A2t_1$ | ... | $Akt_1$ |
| $A1t_2$ | $A2t_2$ | ... | $Akt_2$ |
| ... | ... | ... | ... |
| $A1t_{n1}$ | $A2t_{n2}$ | ... | $Akt_{nk}$ |
| $CentrInt1$ of $Set1$ | $CentrInt2$ of $Set2$ | ... | $CentrIntk$ of $Setk$ |

An intelligence indicator should make a quantitative indication of a system's intelligence. In the case of a particular set of cooperative multiagent systems, the researcher who wishes to compare the intelligence of more multiagent systems should decide on the most appropriate intelligence indicator. Our metric, presented in the form of the *MetrIntSimil* algorithm, is appropriate for multiagent systems, where the problem solving intelligence indicator of each system can be expressed as a single value. If necessary, in the case of a multiagent system, this value can be calculated as the weighted sum of some other values that measure different aspects of the system intelligence Equation (1):

$$IntInd = wg_1 \times ms_1 + wg_2 \times ms_2 + \cdots + wg_r \times ms_r; wg_1 + wg_2 + \cdots + wg_r = 1. \quad (1)$$

Equation (1) indicates the general case when the intelligence indicator is calculated as the weighted sum of $r$ intelligence components measure, where: $ms_1, ms_2, \ldots, ms_r$ denote the intelligence components measure, which are obtained as a result of a problem solving intelligence evaluation; and $wg_1, wg_2, \ldots, wg_r$ denote the intelligence components weights. For illustrative purposes, we present the scenario of an intelligent cooperative multiagent system composed of flying drones (drones with agent properties) denoted *CoopIntDrones*. The drones should cooperatively perform different missions established by a human specialist(s) denoted *HE*. Based on the efficient cooperative solving of difficult problems, the intelligence can be considered at the system's level. The intelligence of such a system cannot be unanimously defined. The human specialist(s) who would like to measure the *CoopIntDrones* intelligence must clarify what he/she understands by intelligence, establish the corresponding problem solving intelligence indicator, and the intelligence components based on that should generate the intelligence indicator. *HE* could consider, for example, the machine intelligence based on the intelligence of fulfilling the mission and the ability to learn. *CoopIntDrones* can learn new data/information/knowledge that could increase the efficiency of cooperation and improve the fulfilling of future missions. As intelligence components, the following could be considered: the necessary time for the fulfilling of the mission; the mission fulfilling accuracy; quantity of new data/information/knowledge learnt at the system's level; quantity of measurable improvement in cooperation efficiency by learning; degree of autonomy in the fulfilling of the mission (counting the number of times for which the remote intervention of human specialists was necessary) and some others.

$CentrInt_1, CentrInt_2, \ldots, CentrInt_k$ represent the central intelligence indicators of the $MA_1$, $MA_2, \ldots, MA_k$. We considered the central intelligence indicators of $MA_1, MA_2, \ldots, MA_k$ as the means or the medians of the $Set1, Set2, \ldots, Setk$. The decision for opting as central intelligence indicator for the *mean* is in the parametric case (all the intelligence indicator data sampled from Gaussian population with equal *Variance* and *Standard Deviation(SD)*; *Variance*=$SD^2$) or the *median* in the nonparametric case (not all the intelligence indicator data sampled from Gaussian population or all the intelligence indicator data sampled from Gaussian population, but not all the intelligence indicator data variances are equal).

Figure 1 presents the flowchart of the main processing steps performed by the *MetrIntSimil* metric. The *MetrIntSimil* algorithm described in detail in Figure 2 compares the intelligence of $MA_1$, $MA_2, \ldots, MA_k$ on some testing problems sets. It checks if the results (more concretely, the central intelligence indicators) are similar or different from a statistical point of view. We call in the following, *Null Hypothesis* denoted as *H0*, the statement that $MA_1, MA_2, \ldots, MA_k$ intelligence are similar from the statistical point of view. We denote by *H1* the *Alternative Hypothesis*, which indicates that the intelligence of $MA_1, MA_2, \ldots, MA_k$ is different from the statistical point of view.

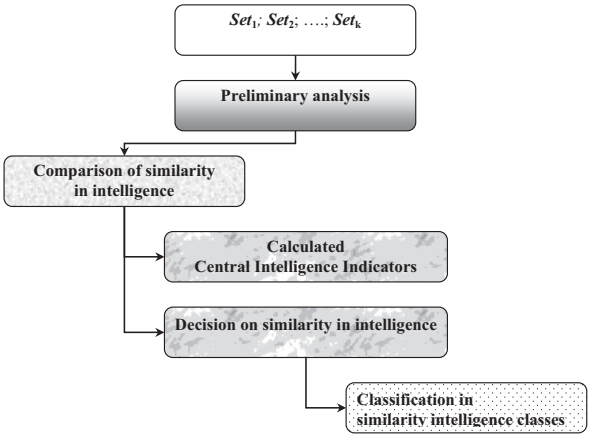



**Figure 1.** The flowchart of the processing performed by the MetrIntSimil metric.

The *MetrIntSimil* metric uses as input $Set1=\{A1t_1, A1t_2, \ldots, A1t_{n1}\}$; $Set2 = \{A2t_1, A2t_2, \ldots, A2t_{n2}\}$; …; $Setk = \{Akt_1, Akt_2, \ldots, Akt_{nk}\}$ that represents the $MA_1$, $MA_2$, …, $MA_k$ intelligence indicators obtained during the *MasSet* intelligence evaluation in solving some sets of test problems.

For the normality verification, we propose the *One-Sample Kolmogorov–Smirnov Goodness-of-Fit test* [52–54] and the *Lilliefors test* [53–55] that is based on the *Kolmogorov–Smirnov test*. For the verification of equality of variances of two samples, the *F test* [56] can be used. For the verification of equality of variances of more than two samples, we propose the use of the *Bartlett test* [57,58].

We propose in some cases the use of a method for the elimination of outlier intelligence indicator values. We call outlier intelligence indicator value, a very high or very low intelligence value, different from those other intelligence indicator values. The difference of an intelligence indicator value from others should be considered from the statistical point of view. Based on this fact, we consider appropriate the application of statistical tests for outlier intelligence values detection. There are many tests for statistical outliers' detection described in scientific literature, like: *Chauvenet's criterion* [59,60], *Peirce's criterion* [61], *Dixon's Q test* [62] and *Grubbs test* [63].

We chose the *Grubbs test* for outliers' detection and decided to apply the significance level $\alpha\_grubbs = 0.05$. At first application, the *Grubbs test* is able to detect a single outlier (if there is at least one outlier). If a value is identified as an outlier, then it can be concluded that this is the most statistically different value from those other measured intelligence indicator values. If an outlier is identified, then a decision of whether the outliers' detection test should be applied again may be considered. This is a recursive process, and the detection method could be applied consecutively more times until there are no other outliers identified.

If sample intelligence data does not follow a Gaussian distribution, then one can opt for the application of a transformation. Some of the most common normalizing transformations are indicated in Table 2 [64].

**Table 2.** Examples of transformation that can be applied in order to obtain normally distributed data.

| Type of Data and Distribution | Normalizing Transformation |
|---|---|
| *Set* comes from Poisson distribution | Square root of *Set* |
| *Set* comes from Binomial distribution | Arcsine of square root of *Set* |
| *Set* comes from Lognormal distribution | Log (*Set*) |

***MetrIntSimil: Multiagent Systems Intelligence Comparison Algorithm***
**IN:***Set1:*={$A1t_1$, $A1t_2$,…, $A1t_{n1}$}; *Set2:*={$A2t_1$, $A2t_2$,..., $A2t_{n2}$};…; *Setk:*={$Akt_1$, $Akt_2$,.., $Akt_{nk}$};
**OUT:***//The calculated Central Intelligence Indicators* of $MA_1$, $MA_2$,..., $MA_k$.
*CentrInt1, CentrInt2,..., CentrIntk;*
*DecisionIntelligence;* //The decision related to the intelligence comparison of $MA_1$, $MA_2$,..., $MA_k$.
**Step1. Preliminary analysis**
@Make a Statistical Characterization of *Set1, Set2,..., Setk;*
@Calculate the *Coefficient of Variation* (*CV*) for *Set1, Set2,...., Setk;*
@Analyze the homogeneity/heterogeneity of *Set1, Set2,..., Setk* based on the *CV* values;
@Verify if *Set1, Set2,..., Setk* are normally distributed;
@If necessary opt for the detection of intelligence outlier values in *Set1, Set2,..., Setk;*
***If*** (it was decided for the application of the intelligence indicators outliers detection) ***then***
    @Eliminate the outliers;
    @Verify if *Set1, Set2,..., Setk* are normally distributed;
***EndIf***
**Step2. Comparison of similarity in intelligence**
//*Normality* verification for *Set1, Set2,..., Setk;*
***If*** (*Set1* **and** *Set2 .....* **and** *Setk* are normally distributed) ***then*** *Normality:=YES;*
    **Else** *Normality:=NO;*
***EndIf***
***If*** (*Normality = YES)* ***then***
*//*Verification of equality of *variances* of *Set1, Set2,..., Setk* using the *Bartlett test.*
@Formulate *H0Var;* //*H0Var* - denotes the *Null Hypothesis* related with the equality of variances;
@Formulate *H1Var;* //*H1Var* - denotes the *Alternative Hypothesis* related with the equality of variances;
@Verify *H0Var* by applying the *Bartlett test*;
***EndIf***
@Formulate *H0* and *H1;//Related to the similarity in intelligence*
***If*** (*Normality=YES* **and** *"H0Var is true"*) ***then***
    //The central intelligence indicators are calculated as the means of *Set1, Set2,..., Setk;*
    @Calculates *CentrInt1, CentrInt2,..., CentrIntk* as the means of *Set1, Set2,..., Setk;*
    @Apply the *Single-Factor ANOVA* test at the chosen significance level *α_int*;
    @As a test result will be obtained the *P-Value*;
***EndIf***
***If*** **((***Normality*=NO) **or** ((*Normality*=YES) **and** ("*H1Var is true"*))) ***then***
    //The central intelligence indicators are calculated as the medians of *Set1, Set2,..., Setk.*
    @Calculates *CentrInt1, CentrInt2,..., CentrIntk* as the medians of *Set1, Set2,..., Setk;*
    @Apply the *Kruskal-Wallis* test at the chosen significance level *α_int*;
    @As a test result will be obtained the *P-Value*;
***EndIf***
**Step3. Presentation of the Central Intelligence Indicators**
@Report *CentrInt1, CentrInt2,…*, *CentrIntk*;
**Step4. Classification in similarity intelligence classes**
***If*** (*P-Value*>α_int) ***then***
    @Accept *H0*
    *DecisionIntelligence="*$MA_1$, $MA_2$, ....$MA_k$ *intelligences are similar"*
        ***Else***
        @Accept *H1* //*H0* is rejected.
        *DecisionIntelligence="*$MA_1$, $MA_2$,..., $MA_k$ *intelligences are different"*
        @Makes the classification in intelligence classes.
        ***If*** (was applied the *Kruskal-Wallis* test) ***then***
        @It is applied the *Dunn test* to compare all the selected intelligence indicator data sets (the central intelligence indicators) in pairs.
        ***Else***
        @It is applied the *Tukey test* to compare all the intelligence indicator data sets (the central intelligence indicators) in pairs.
        ***EndIf***
***EndIF***
***EndMultiagentSystemsIntelligenceComparison***

**Figure 2.** The proposed MetrIntSimil metric.

For the effective comparison of intelligence of the multiagent systems, the parametric *Single-Factor ANOVA test* [65] or the nonparametric *Kruskal–Wallis test* [66] should be applied. In case of choosing each of them, the $\alpha\_int$ (the significance levels at which the statistical test is applied) value should be established. We suggest as the value of $\alpha\_int$, $\alpha\_int = 0.05$, which we consider as the most appropriate. $\alpha\_int$ denotes the probability to make a Type I error, to reject *H0* (Null Hypothesis) when it is true. A Type I error means detecting an effect that is not present.

In the proposed metric algorithm, if $p$-value $> \alpha\_int$ ($p$-value obtained by applying the *Single-Factor ANOVA test* or the *Kruskal–Wallis test*), then it can be decided that *H0* could be accepted. The conclusion states that even if there is a numerical difference between the calculated central intelligence indicators $CentrInt1, CentrInt2, \ldots, CentrIntk$, there is no statistical difference between the intelligence of the studied $k$ multiagent systems. The numerical difference is the result of the variability in the intelligence of the multiagent systems. In this situation, from the classification point of view, all of the multiagent systems $MA_1, MA_2, \ldots, MA_k$ can be classified in the same class composed of systems with similar intelligence.

If *H1* is accepted (as result of *H0* rejection), then the intelligence level of $MA_1, MA_2, \ldots, MA_k$ is different. The numerical difference between the central intelligence indicators $CentrInt1, CentrInt2, \ldots, CentrIntk$ is statistically significant and is not the consequence of the variability. From the classification point of view $MA_1, MA_2, \ldots,$ *and* $MA_k$ cannot be classified in the same class composed of systems with similar intelligence. If $H1$ is accepted, then the *Dunn test* [67] or *Tukey test* [68,69] should be applied, which allow the classification in intelligence classes of all the studied *CMASs*. More concretely, these tests make a statistical comparison between the central intelligence indicators, the mean in parametric case or the median in the nonparametric case.

*Tukey test* [68,69] is a single-step parametric multiple pairwise comparison method. *Tukey test* can be used as a post hoc analysis following the rejection of *Single-Factor ANOVA test* null hypothesis.

*Dunn's test* [67] is a non-parametric multiple pairwise comparisons method. *Dunn's test* is based on rank sums. It is used as a post hoc method following rejection of a *Kruskal–Wallis test* null hypothesis. In the case study presented in a further section based on the non-parametric data, the *Dunn test* is applied.

## 4. Measuring the Intelligence of More CMASs—A Case Study on Solving the Symmetric TSP

### *4.1. Symmetric Travelling Salesman Problem Solving*

The *TSP* has many applications, such as logistics, planning, and the manufacture of microchips. A sub-problem in DNA sequencing is represented as a slightly modified version of the *TSP* [70]. The concept city represents DNA fragments, and the concept distance represents a similarity measure between DNA fragments. Ref. [71] analyses the computational recognition of RNA Splice Sites described as a modified *TSP* a *Quadratic Traveling Salesman Problem* by some exact algorithms. Ref. [72] presents an evolution based biosensor receptor DNA Sequence generation algorithm. The *TSP* approach is applied in the proposed algorithm to evaluate the safety and fitness of the generated DNA sequences.

The *Travelling Purchaser Problem (TPP)* and the *Vehicle Routing Problem (VRP)* are both NP-hard problems that represent generalizations of *TSP*. Since *TSP* is NP-hard, *TPP* and *VRP* are also NP-hard. The *TPP* can be enounced as follows "Given a list of marketplaces, the cost of travelling between different marketplaces, and a list of available goods together with the price of each such good at each marketplace, the task is to find, for a given list of articles, the route with the minimum combined cost of purchases and traveling" [73]. The *VRP* can be enounced as follows "What is the optimal set of routes for a fleet of vehicles to traverse in order to deliver to a given set of customers?" [74].

Both variants of the *TSP*, the *Asymmetric TSP (ATSP)* and the *Symmetric TSP (STSP)* are frequently studied. The *ATSP* characterizes the situation when edges may not exist in both directions or the distances might be different; a directed graph is formed. Traffic collisions, one-way streets, are examples of situations when the symmetry property is not satisfied. Dantzig et al. [75] formulated the asymmetric

traveling salesman problem as a 0–1 linear program on a graph (*V*,*E*). Their formulation for the symmetric case gives rise to the standard subtour elimination polytope *SEP*(*n*).

The *STSP* [11] is the problem of finding the shortest Hamiltonian cycle/tour in a weighted undirected graph (the distance between two cities is the same in each opposite direction) that does not have loops or multiple edges. Many practical combinatorial optimization problems in production management and scheduling can be formulated as equivalent to the *STSP*. The symmetry property is useful based on the fact that it halves the number of possible solutions. There are many studies and research performed on the *STSP* [76–78]. Ref. [79] proposes a transforming of asymmetric into symmetric TSP.

Ref. [80] proposes an alternate formulation of the symmetric traveling salesman problem and presents its properties. The polytope defined by this formulation, $U(n)$, is compared with the standard subtour elimination polytope $SEP(n)$. It is proved that $U(n) \subseteq SEP(n)$.

Ref. [81] proposes a novel lower bounding and reduction procedures for the *STSP*. In the proposal, the lower bounds are obtained through the solution of the linear program corresponding to the 2-matching relaxation, being improved through a restricted Lagrangean 2-matching approach. It presents a comparison of the new bounds with bounds obtained with the well-known Lagrangean 1-tree relaxation.

### *4.2. CMASs That Operate by Mimicking Biological Ants*

The ability of biological ants to determine shortest paths to food was studied in many research papers [82–84]. An individual ant possesses few capabilities, but, when operating in swarms/colonies, the ants are capable of having a complex surviving behavior. Biological ants communicate by pheromones, which allow the finding of the shortest path between food sources and their nests. All ants deposit pheromones on the trail while walking. Each ant follows the pheromone trail that it meets with some probability, and which is proportional to the density of the pheromone. The more ants walk on a trail, the more pheromone is deposited on it, and the more and more ants follow that trail. Based on this collaboration mechanism, there is a very high probability for the ants to find a very short path (shortest path or close to it).

For proving the effectiveness of the designed *MetrIntSimil* metric, we conducted a case study. The *Symmetric Travelling Salesman Problem* solved by three *CMASs* formed by autonomous mobile agents that mimic biological ants was considered. We understand by mobile agents simple computing embedded agents able to move in the environment during the problem solving. The communication between the agents is carried out by signs and it is similar to the communication of the biological ants by using pheromones. This type of communication allows an efficient, flexible and robust cooperative problem solving even in *CMASs* composed of a large number of individuals. There are many studies conducted on systems composed of artificial ants and their applications for different problem solving [85,86].

Marco Dorigo in his Ph.D. thesis [16,17] proposed first the problem solving based on simple computing agents that mimic the operation of biological ants. Artificial ants (operating as reactive agents known in the intelligent agent literature) imitate the behavior of biological ants on how they search for food. As a general idea, in an Ant System, initially, each agent is placed on a randomly chosen city (node of the graph). An agent $k$ currently at node $i$ chooses to move to node $j$ by applying the following probabilistic transition rule Equation (2). After each agent completes its tour, the pheromone amount on each path will be adjusted according to Equations (3)–(5):

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta}}{\sum_{l \in J_k(i)} [\tau_{il}(t)]^{\alpha} \times [\eta_{il}]^{\beta}}, & \text{if } j \in J_k(i), \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t), \tag{3}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t), \tag{4}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{L_k}, & \text{If } (i,j) \ \epsilon \text{ tour down by agent } k, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

In Formulas (2)–(5), the following notations are used: $\alpha$ and $\beta$ are adjustable parameters. $\alpha$ and $\beta$ control the relative weights of the heuristic visibility and the pheromone trail. In the parameters establishment, a trade-off between edge length and pheromone intensity appears to be necessary. $Q$ denotes an arbitrary constant. $d_{k,h}$ represents the distance between the nodes ($k$ and $h$); $\eta_{kh}$, $\eta_{kh} = 1/d_{k,h}$ the heuristic visibility of edge $(k,h)$. 1-$\rho$ is the pheromone decay parameter, $0 < \rho < 1$, and it represents the trail evaporation when the agent chooses a node where it decides to move. $m$ denotes the number of agents, and $L_k$ denotes the length of the tour performed by agent $k$.

### *4.3. The Experimental Setup*

For the validation of the proposed metric, we carried out experimental evaluations for different sets of cooperative multiagent systems. We experimented: *Ant System, Elitist Ant System, Ranked Ant System, Best-Worst Ant System, Min-Max Ant System* and *Ant Colony System*. Finally, we decided to present the validation for three cooperative multiagent systems between which the central intelligence indicator value does not have a high difference. In the experiment that we present in the following, we considered three cooperative multiagent systems denoted as $MA_1$, $MA_2$ and $MA_3$, which operated as a *Best-Worst Ant System* [12,13] the $MA_1$; a *Min-Max Ant System* [14,15] the $MA_2$ and *Ant Colony System* [16,17] the $MA_3$. All of them were applied for the *Symmetric Travelling Salesman Problem* solving. *STSP* is a well known NP-hard problem [8]. In each multiagent system, the agents cooperated in order to efficiently solve the undertaken problems by the multiagent system. $MasSet$=$\{MA_1, MA_2, MA_3\}$. $|MasSet| = k = 3$.

In the experimental setup, for all the analyzed multiagent systems, we considered maps with $nr$ = 35 randomly placed cities on the map. All the studied multiagent systems were composed of $m$ = 10 cooperative reactive agents. As parameters, all the cooperative multiagent systems were considered: *Number Of Tests* = 1000; $\alpha$ (power of the pheromone); $\beta$ (power of the distance/edge weight); *evaporation* (the evaporation factor). We choose as parameter values (experimentally established): $\alpha$ = 1; $\beta$ = 1 and *evaporation* = 0.1. Table 3 presents the obtained simulation results. In the simulations, the obtained best to date travel value from the end of the problem solving was considered for the case of all multiagent systems as the intelligence indicator.

**Table 3.** The obtained Intelligence Indicators for $MA_1$, $MA_2$ and $MA_3$.

| ***Set1*/$MA_1$** | ***Set2*/$MA_2$** | ***Set3*/$MA_3$** |
|---|---|---|
| 7.172; 6.864; 7.691; 6.12; 6.572; 6.612; 7.413; 5.786; 6.262; 6.626; 6.135; 6.217; 6.586; 6.467; 8.084; 7.313; 7.295; 6.473; 6.516; 6.657; 7.009; 8.297; 7.714; 6.729; 7.177; 6.887; 6.612; 5.99; 7.007; 7.333; 5.78; 6.585; 7.257; 7.225; 8.005; 6.592; 6.741; 6.37; 5.944; 6.573; 6.911; 6.513; 7.447; 7.066; 7.277; 6.924; 7.343; 6.204 | 5.657;5.706; 5.409; 5.442; 5.826; 5.123; 4.81; 5.579; 5.853; 5.121; 5.459; 4.492; 4.944; 5.466; 4.978; 5.095; 5.917; 5.76; 5.315; 5.558; 5.661; 5.546; 5.809; 5.729; 5.519; 5.288; 5.293; 5.365; 5.806; 4.465; 5.427; 5.217; 5.244; 5.741; 5.724; 5.579; 5.599; 5.506; 5.907; 5.421; 5.312; 5.632; 5.101; 5.476; 4.405; 5.932; 5.454; 5.065; 5.111; 5.345 | 5.342; 4.868; 6.134; 5.251; 4.85; 5.307; 5.605; 5.624; 6.053; 4.788; 5.055; 5.153; 5.779; 4.87; 5.339; 4.992; 5.322; 5.363; 5.493; 5.004; 5.261; 5.476; 5.469; 7.278*; 5.53; 5.084; 5.337; 5.595; 5.352; 4.631; 5.068; 4.911; 4.831; 5.431; 4.933; 4.987; 5.092; 5.377; 5.589; 5.623; 5.563; 5.195; 5.926; 5.136; 5.325; |

Table 4 presents the obtained results by analyzing *Set*1, *Set*2, and *Set*3. In the table, the following notations were used: *SEM* denotes the standard error of the mean; [*LCI95%*, *UCI95%*] denotes the 95% confidence interval of the mean; *Lowest* denotes the smallest; *Highest* denotes the utmost; *K-S Stat* denotes the Kolmogorov–Smirnov test statistic; *Lill Stat* denotes the Lillefors test statistic. The normality was verified by using the *One-Sample Kolmogorov–Smirnov Goodness-of-Fit test* and *Lilliefors test*, applied at $\alpha\_normal = 0.05$ significance level. It can be noticed that the *Set*3 does not pass the normality test. In order to obtain normally distributed data on *Set*3, we applied the *Grubbs test* with $\alpha\_grubbs = 0.05$. The *Grubbs test* identified the value 7.278 as a single outlier. It was identified at the first application of the test. When applying the test again, no outlier was detected. We calculated $Set3^* = Set3 - \{7.278\}$.

The last column of Table 4 presents the analysis results for *Set*3*. Now, the *Set*3* passed the normality test. For the verification of the equality of standard deviations of *Set*1, *Set*2, and *Set*3*, we applied the *Bartlett test* at the $\alpha\_bart = 0.05$ significance level. The obtained *p*-value (*p*-value of *Bartlett's test*) was *p*-value = 0.0002 (*p*-value $< \alpha\_bart$), which suggested that the difference between the standard deviations of *Set*1, *Set*2, and *Set*3* is very significant. Another calculation detail was the obtained Bartlett statistics value 17.225.

**Table 4.** Results of the intelligence indicator sample data analysis.

| **Type of Analysis** | $MA_1/Set1$ | $MA_2/Set2$ | $MA_3/Set3$ | $MA_3/Set3^*$ |
|---|---|---|---|---|
| *Mean* | 6.841104 | 5.40378 | 5.3376 | 5.2935 |
| *SD/Variance* | 0.5861/0.3435 | 0.3647/0.133 | 0.4471/0.1999 | 0.3392/0.1151 |
| *Sample size* | 48 | 50 | 45 | 44 |
| *SEM* | 0.08460 | 0.05158 | 0.06666 | 0.05113 |
| *[LCI95%, UCI95%]* | [6.671, 7.011] | [5.3, 5.508] | [5.203, 5.472] | [5.19, 5.397] |
| *Lowest/Highest* | 5.780/8.297 | 4.405/5.932 | 4.631/7.278 | 4.631/6.134 |
| *Median* | 6.735 | 5.454 | 5.325 | 5.324 |
| *CV* | ≈8.5673 | ≈6.7499 | ≈8.3764 | ≈6.4079 |
| *K-S Stat/p-value* | 0.1024/>0.1 | 0.1057/>0.1 | 0.1498/0.0128 | 0.07404/>0.1 |
| *Lill Stat/p-value* | 0.102/0.2 | 0.106/0.2 | 0.15/0.013 | 0.074/0.2 |
| *Normality passed* | YES | YES | NO | YES |

Figure 3 contains the graphical representation of intelligence indicators, *Set*1, *Set*2, and *Set*3, with all the intelligence indicator data included. As an observation, we mention that the data was not paired. We understand by pairing the fact that the experimental intelligence evaluation number 1 for $MA_1$, $MA_2$ and $MA_3$ was not carried out for the same problems. The experimental intelligence evaluation number 2 for $MA_1$, $MA_2$ and $MA_3$ was not carried out for the same problem. The lines represented in Figure 3 illustrate the intelligence variation in problem solving. The *MetrIntSimil* metric algorithm does not restrict the number of intelligence evaluations for all of the multiagent systems to be the same. In our experimental setup, there were $|Set1| = 48$, $|Set2| = 50$ and $|Set3| = 45$ considered experimental evaluations of the intelligence. Figure 4 is similar to content of Figure 3, but it contains the graphical representation of the obtained intelligence indicators data with the outlier intelligence values excluded.

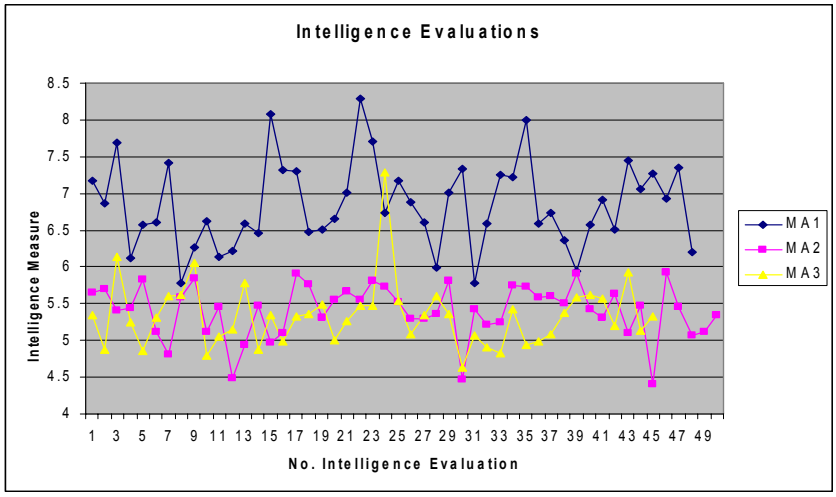



**Figure 3.** Graphical representation of the intelligence indicators of $MA_1(Set1)$, $MA_2(Set2)$, and $MA_3(Set3)$.

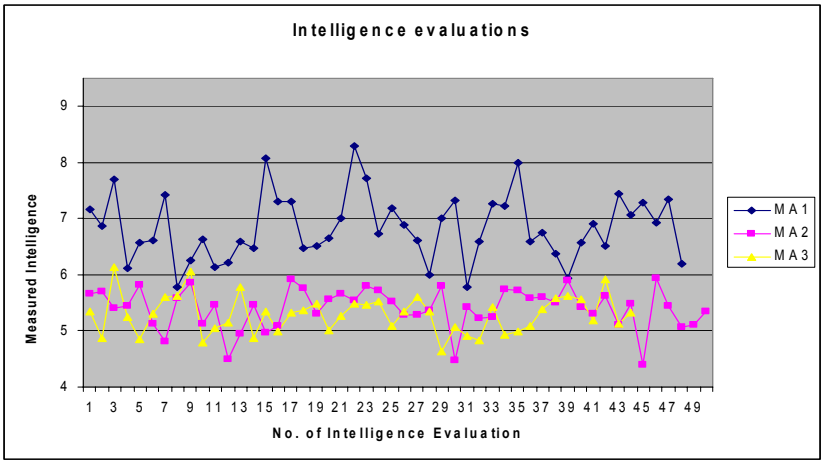



**Figure 4.** Graphical representation of the intelligence indicators of $MA_1(Set1)$, $MA_2(Set2)$, and $MA_3(Set3^*)$ with the outliers excluded.

The *Coefficient of Variation (CV)* of a sample intelligence indicator data should be calculated using formula 6. In formula 6, the *SD/Mean* is multiplied by 100 for obtaining the result in percentages. For example, *SD/Mean* = 0.3 multiplied by 100 gives 30. We used the *CV* value for analyzing the homogeneity–heterogeneity of the intelligence indicator data [87,88]. We consider the data classification based on the variability as follows. A classification of the homogeneity–heterogeneity of the data can be carried out based on the $CV$. $CV \in [0, 10)$ indicates homogeneous data; $CV$, $CV \in [10, 30)$ indicates relatively homogeneous data; $CV$, $CV \geq 30$ indicates heterogeneous data:

$$CV = 100 \times (SD/Mean). \tag{6}$$

Based on the obtained results, all the samples *Set*1, *Set*2 and *Set*3* were normally distributed, but having different variances; the *MetrIntSimil* algorithm indicated the application of the nonparametric *Kruskal–Wallis test* with $\alpha\_int = 0.05$. $p$-value $\approx 0.0001$, obtained as a result of the *Kruskal–Wallis*

*test*, *p*-value $<$ $\alpha\_int$. Based on this result, it should be concluded that *H0* cannot be accepted. The intelligence level of all the studied multiagent systems $MA_1$, $MA_2$, $MA_3$ cannot be considered to be the same. From the classification point of view, the multiagent systems cannot be classified in the same intelligence class.

According to the *MetrIntSimil* algorithm, as a next step, we applied the *Dunn test* [67] for all the pairs of *Set*1, *Set*2, *Set*3*, with significance level $\alpha\_dunn = 0.05$. The obtained results, presented in Table 5, prove that $MA_2$ and $MA_3$ can be included in the same class of systems with similar intelligence denoted in the following $IntClassB$. $MA_1$ should be included in a separate class of systems with similar intelligence denoted $IntClassA$. The multiagent systems that belongs to $IntClassB$ has a higher intelligence than the multiagent system that belongs to the $IntClassA$ intelligence class. As an intelligence indicator in our study, we have considered the global-best, this having the significance that a lower value indicates higher intelligence. For example, the global-best by 4.868 is better than 8.084.

**Table 5.** Results of the Dunn test for paired intelligence comparison.

| Compared CMASs | *p*-Value | Mean Rank Difference | Interpretation of the Result |
|---|---|---|---|
| $MA_1$ vs. $MA_2$ | <0.001 | 64.909 | $MA_1$ and $MA_2$ * |
| $MA_1$ vs. $MA_3$ | <0.001 | 76.241 | $MA_1$ and $MA_3$ * |
| $MA_2$ vs. $MA_3$ | >0.05 | 11.331 | $MA_2$ and $MA_3$ ** |

* CANNOT be considered to belong to the same similarity intelligence class; ** CAN be considered to belong to the same similarity intelligence class.

The obtained results prove (Table 5, the comparison result of $MA_2$ vs. $MA_3$) that $MA_2$ and $MA_3$ can be included in the same similarity class of intelligence, denoted in the following by $IntClassB$. As another proof of this fact, the metric algorithm can be applied again just for the $MA_2$ characterized by the *Set*2 of intelligence indicators and $MA_3$ characterized by the *Set*3* of intelligence indicators. Table 4 presents the fact that the normality test for *Set*2 and *Set*3* was passed. For the verification of the equality of variances of *Set*2 and *Set*3*, we applied the *F-test* at significance level $\alpha\_f = 0.05$. The result of the *F-test* is *p*-value = 0.63. *p*-value > $\alpha\_f$, which indicates that the difference between the variances of *Set*2 and *Set*3* is not statistically significant. Based on these considerations, the application of the *Single-Factor ANOVA test* at significance level $\alpha\_int$ = 0.05 can be considered. The obtained result was *p*-value = 0.1341. *p*-value$>$ $\alpha\_int$, sustaining the conclusion that the two multiagent $MA_2$ and $MA_3$ can be included in the same similarity intelligence class.

## 5. Discussion and Comparison of the MetrIntSimil Metric

In our research, we considered the difficult problem solving intelligence measuring at the level of the whole cooperative multiagent system, not at the individual/agent level. Our metric, presented in the form of the algorithm *MetrIntSimil*, is appropriate for multiagent systems, where the intelligence indicator of a problem solving by a multiagent system can be expressed as a single value. If necessary, this value can be calculated as the weighted sum of some values of more intelligent components that measure different aspects of the system intelligence.

An intelligence indicator should make a quantitative indication of a system intelligence in solving a difficult problem. The researcher who wishes to make a comparison of the intelligence of two or more multiagent systems should decide on the type of intelligence indicator. For all the compared multiagent systems, the type of intelligence indicator should be the same. We consider that an effective metric must be able to measure the same type of intelligence. As an example, in the case of biological systems, it makes no sense to compare the intelligence of a fish with the intelligence of a bird.

The elaborated metric takes into consideration the variability in the intelligence of the compared multiagent systems. A multiagent system could have different intelligent reactions in different situations. In a specific situation, the reaction could be more or less intelligent. In our research, we considered the presence of high and low outlier intelligence values, which are statistically very

different from all other intelligence values. If such outlier values are taken into consideration, this could influence the comparison result of more multiagent systems' intelligence.

In our research, we considered the necessity of the establishment of a central intelligence indicator of a *CMAS*, which illustrates the central intelligence tendency of the multiagent system. We considered as possible central intelligence indicators, and the calculation as the means of intelligence indicators sample data in the parametric case (all *Set*1, *Set*2, . . . , *Setk* are sampled from a Gaussian population and their variance is equal from statistical point of view) and as the medians in the nonparametric case (not all the intelligence indicators data are sampled from Gaussian population or all the intelligence indicator data sampled from Gaussian population, but they have different variances). The median is more robust than the mean, a higher or lower value influences more the mean than the median.

We did not find in the scientific literature an effective metric based on difficult problem solving intelligence measuring that has all the properties of *MetrIntSimil* metric, such as: allowing the simultaneous intelligence comparison of two or more than two multiagent systems; and accuracy and robustness in comparison and universality at the same time.

*MetrIntComp* metric [51] presented in the scientific literature is able to make a comparison of two cooperative multiagent systems' intelligence. The *MetrIntComp* metric is based on a similar principle of difficult problem solving intelligence measuring as the *MetrIntSimil* metric. The *MetrIntComp* metric uses difficult problem solving intelligence evaluation data, based on which it makes a mathematically grounded comparison of exactly two cooperative multiagent systems intelligence. This allows the classification of the compared systems in intelligence classes (classification in the same class or in different classes). The main advantage of the *MetrIntComp* metric is the robustness. The robustness is assured by the fact that in the metric algorithm for the obtained intelligence indicator data comparison, the two unpaired samples *Mann–Whitney test* is used that is known as a nonparametric robust test [89,90]. It does not require data normality (that the samples belong to a Gaussian distribution).

*MetrIntSimil* based on the obtained intelligence indicators makes a mathematically grounded analysis. At a specific step of the *MetrIntSimil* metric algorithm based on some analysis, it chooses between the application of the parametric *Single-Factor ANOVA* test [65] and nonparametric *Kruskal–Wallis* test [66]. Based on this fact, the *MetrIntSimil* metric is accurate and robust at the same time. *MetrIntSimil* conserves and extends the properties and advantages of the *MetrIntComp* metric. In the case of normally distributed intelligence indicator data with same variances, *MetrIntSimil* is able to apply a parametric test that is the most appropriate. Another advantage consists in the necessary sample size of intelligence indicators. If a parametric test could be applied, then the required sample size should be smaller than in the nonparametric case.

The *Mann–Whitney test* for two unpaired samples is the non-parametric analog to the two-sample unpaired *t*-test. It uses a different test statistic comparatively with the *Kruskal–Wallis* test (*U* instead of the *H* of the *Kruskal–Wallis test*), but the *p*-value is mathematically identical to that of a *Kruskal–Wallis test* [91,92].

Comparatively with the *MetrIntComp* metric, the *MetrIntSimil* metric is able to make a simultaneous comparison of more than two multiagent systems, with the established significance level $\alpha\_int$ (the probability of making a Type I error is $\alpha\_int$). *MetrIntComp* could be used for the comparison of more that two cooperative multiagent systems, pair-by-pair, but this approach is not appropriate. The probability of making a Type I error increases as the number of tests increase. If the significance level is set at $\alpha$, the probability of a Type I error can be obtained, regardless of the number of groups being compared. For example, if the probability of a Type I error for the analysis is set at $\alpha = 0.05$ and six two-sample tests (*t*-test for example) are performed, the overall probability of a Type I error for the set of tests $\alpha\_overall = 1–0.95^6 = 0.265$.

In the scientific literature, there is no universal view on what intelligence metrics should measure. Each of the designed metrics consider the machine intelligence based on different principles. Based on this fact, most of them cannot be effectively compared directly with each other. For comparison reasons,

we chose a recent intelligence metric called *MetrIntComp* [51], which made possible the comparison and this opens a research direction to standardization of the intelligence metrics. Table 6 summarises the comparison results.

**Table 6.** Comparison of the MetrIntSimil metric with the MetrIntComp metric.

| Comparison criterion | *MetrIntComp* | *MetrIntSimil* |
|---|---|---|
| Principle | * | * |
| Applicability | @ | @ |
| Computations | + | + |
| Number of compared CMASs | 2 | Any number |
| Properties | Universality, Robustness | Universality, Robustness, Accuracy |
| Variability in intelligence | Treated | Treated |

* Intelligence in solving difficult problems; @ Choosing of the system able to solve most intelligently difficult problems. + (1) Calculus of a machine intelligence measure; (2) Verification of the similarity in intelligence; (3) Classification in similarity intelligence classes.

A case study was realized for the experimental evaluation of the *MetrIntComp* metric proposed in the paper [51]. It measured and compared the intelligence of two cooperative multiagent systems in solving an NP-hard problem, the *Symmetric TSP* more concretely. We used the intelligence indicators reported in the paper [51] and applied on them the *MetrIntSimil* metric. The same result was obtained for *MetrIntSimil* as was obtained by applying the *MetrIntComp* metric. Both of the metrics made a differentiation in intelligence between the two studied cooperative multiagent systems, even if the numerical difference between the measured intelligence was small. Based on this fact, the two multiagent systems could not be considered to belong to the same class of intelligence and should be classified in different classes of intelligence.

## 6. Theory of Multiple Intelligences in Machines: The Next Research Works

Humans can solve more or less intelligent problems that require different types of thinking [93]: musical-rhythmic, visual-spatial, verbal-linguistic, logical-mathematical, bodily-kinesthetic, interpersonal, intrapersonal and naturalistic. The theory of multiple intelligences proposed by Howard Gardner differentiates intelligence into specific modalities [94,95]. According to the theory of multiple intelligences, the intelligence is not dominated by a single general ability.

In the next research work, we will focus on the development of a theory of multiple intelligences in machines. Our preliminary conclusion is that such a theory should not be designed based on similarity with the theory of multiple intelligences in humans. The human intelligence and the artificial intelligence are not similar. They are of a completely different type. Life on earth is the result of an evolution by 0.5 billion years [96]. We consider that different types of problem solving by intelligent systems require different types of machine intelligence. In this framework, we consider an important particular research direction: the automatic detection of different types of intelligence that an intelligent artificial system possesses.

With illustrative purpose, we mention the scenario of next-generation flying agent-based drones (flying drones with properties of intelligent agents) able to transport passengers. Such developments certainly will be built in the future. Such an intelligent drone must detain different types of intelligence specific to different types of problems and sub-problems. We consider that, in such a scenario, the clear definition of similarity between drones is necessary. Drones could be specialized: in transporting very few passengers, and in transporting a large number of passengers. As examples of types of intelligence that a drone can possess, it can be noticed: intelligence in communication with the passengers; intelligence in communication and cooperation with other similar flying drones; intelligence in communication and cooperation with other non-similar flying drones; intelligence to fly in difficult weather conditions; and intelligence in avoiding different objects during the flight and some others. The intelligence measure has the sense to be compared to similar drones. Based on a specific type of

intelligence, a flying drone could be more intelligent than another similar flying drone. For another type of problem, the situation could be the vice versa. For example, the flying drone *SimFDA* could be more intelligent in the communication with the passengers than the flying drone *SimFDB*. In the avoiding of the objects during the flight, *SimFDB* could be more intelligent than *SimFDA*.

## 7. Conclusions

Intelligent cooperative multiagent systems *(CMASs)* by a large diversity are used for many real life problem solving tasks. There are very few metrics designed for the quantitative evaluation of *CMASs* intelligence. There are even fewer metrics that allow also an effective quantitative comparison of the intelligence level of more multiagent systems. In this paper, we proposed a novel metric called *MetrIntSimil* that allows an accurate and robust symmetric comparison of the similarity in intelligence of two or more than two *CMASs*. The proposed metric efficiently takes into account the variability in the intelligence of the compared *CMASs*.

For validation purposes of the *MetrIntSimil* metric, we conducted a case study for three cooperative multiagent systems, $MA_1$ that operated by mimicking a *Best-Worst Ant System* [12,13], $MA_2$ that operated by mimicking an *Min-Max Ant System* [14,15] and $MA_3$ that operated by mimicking an *Ant Colony System* [16,17]. The evaluation was carried out for solving a NP-hard problem, the *Symmetric Traveling Salesman Problem* [11]. The proposed metric identified that two of the multiagent systems $MA_2$ and $MA_3$ have similar intelligence level, and, based on that, they can be classified in the same similarity class of intelligence denoted $IntClassB$. The multiagent $MA_1$ intelligence is different from the other two multiagent systems intelligence, and, based on that, it should be considered that it belongs to another intelligence class that we denoted by $IntClassA$. Another conclusion consists in the fact that the multiagent systems belonging to $IntClassB$ have a higher intelligence level than those that belong to $IntClassA$.

The universal *MetrIntSimil* metric is not dependent on aspects/details like the studied/compared cooperative multiagent systems' architecture. It could be applied even to comparison of similarity in intelligence of systems that operate individually without cooperating. Based on a comprehensive study of the scientific literature, we consider that our proposed metric is original and will represent the basis for intelligence measuring and comparison of systems intelligence in many future research works worldwide.

**Acknowledgments:** Matthias Dehmer thanks the Austrian Science Funds for supporting this work, P 30031.

**Author Contributions:** Laszlo Barna Iantovics, Matthias Dehmer and Frank Emmert-Streib designed the proposed intelligence metric; Laszlo Barna Iantovics, Matthias Dehmer and Frank Emmert-Streib conceived and designed the experiments; Laszlo Barna Iantovics performed the experiments; Laszlo Barna Iantovics, Matthias Dehmer and Frank Emmert-Streib analyzed the data; Laszlo Barna Iantovics, Matthias Dehmer and Frank Emmert-Streib contributed analysis tools; Laszlo Barna Iantovics, Matthias Dehmer and Frank Emmert-Streib wrote the paper.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Guillaud, A.; Troadec, H.; Benzinou, A.; Bihan, J.L.; Rodin, V. Multiagent System for Edge Detection and Continuity Perception on Fish Otolith Images. *EURASIP J. Appl. Signal Process.* **2002**, *7*, 746–753.
2. Iantovics, L.B. A New Intelligent Mobile Multiagent System. In Proceedings of the IEEE-SOFA 2005, Szeged, Hungary and Arad, Romania, 27–30 August 2005; pp. 153–159.
3. Iantovics, L.B.; Zamfirescu, C.B. ERMS: An Evolutionary Reorganizing Multiagent System. *Innov. Comput. Inf. Control* **2013**, *9*, 1171–1188.
4. Stoean, C.; Stoean, R. *Support Vector Machines and Evolutionary Algorithms for Classification*; Intelligent Systems Reference Library; Springer International Publishing: Cham, Switzerland, 2014; Volume 69, pp. 570–573.
5. Chen, M.H.; Wang, L.; Sun, S.W.; Wang, J.; Xia, C.Y. Evolution of cooperation in the spatial public goods game with adaptive reputation assortment. *Phys. Lett. A* **2016**, *380*, 40–47.

6. Wang, C.; Wang, L.; Wang, J.; Sun, S.; Xia, C. Inferring the reputation enhances the cooperation in the public goods game on interdependent lattices. *Appl. Math. Comput.* **2017**, *293*, 18–29.
7. Yang, K.; Galis, A.; Guo, X.; Liu, D. Rule-Driven Mobile Intelligent Agents for Real-Time Configuration of IP Networks. In *International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, Palade, V., Howlett, R.J., Jain, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; Volume 2773, pp. 921–928.
8. Van Jan, L. (Ed.) Algorithms and Complexity. In *Handbook of Theoretical Computer Science*; Elsevier: Amsterdam, The Netherlands, 1998; Volume A.
9. Crisan, G.C.; Nechita, E.; Palade, V. On the Effect of Adding Nodes to TSP Instances: An Empirical Analysis. In *Advances in Combining Intelligent Methods*; Hatzilygeroudis, I., Palade, V., Prentzas, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2016; Volume 116, pp. 25–45.
10. Karp, R.M. Reducibility among Combinatorial Problems. In *Complexity of Computer Computations*; Miller, R.E., Thatcher, J.W., Eds.; Plenum Press: New York, NY, USA, 1972; pp. 85–103.
11. Grotschel M.; Padberg M.W. On the Symmetric Travelling Salesman Problem: Theory and Computation. In *Optimization and Operations Research*; Henn R., Korte, B., Oettli, W., Eds.; Springer: Berlin/Heidelberg, Germany, 1978; Volume 157.
12. Zhang, Y.; Wang, H.; Zhang, Y.; Chen, Y. Best-Worst Ant System. In Proceedings of the 3rd International Conference on Advanced Computer Control (ICACC), Harbin, China, 18–20 January 2011; pp. 392–395.
13. Cordon, O.; de Viana, I.F.; Herrera, F. Analysis of the Best-Worst Ant System and Its Variants on the QAP. In *International Workshop on Ant Algorithms*; Dorigo, M., Di Caro G., Sampels, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2463.
14. Prakasam, A.; Savarimuthu, N. Metaheuristic algorithms and probabilistic behaviour: A comprehensive analysis of Ant Colony Optimization and its variants. *Artif. Intell. Rev.* **2016**, *45*, 97–130.
15. Stutzlem, T.; Hoos, H.H. Max-Min Ant System. *Future Gener. Comput. Syst.* **2000**, *16*, 889–914.
16. Colorni, A.; Dorigo, M.; Maniezzo, V. Distributed Optimization by Ant Colonies. In *Actes de la Premiere Conference Europeenne sur la vie Artificielle*; Elsevier: Paris, France, 1991; pp. 134–142.
17. Dorigo, M. Optimization, Learning and Natural Algorithms. Ph.D. Thesis, Politecnico di Milano, Milan, Italy, 1992.
18. Lin, Y.; Duan, X.; Zhao, C.; Xu, L. *Systems Science Methodological Approaches*; CRC Press: Boca Raton, FL, USA, 2012.
19. Xu, L. The Contribution of Systems Science to Information Systems Research. *Syst. Res. Behav. Sci.* **2000**, *17*, 105–116.
20. Langley, P.; Laird, J.E.; Rogers, S. Cognitive architectures: Research issues and challenges. *Cogn. Syst. Res.* **2009**, *10* 141–160.
21. Tang, C.; Xu, L.; Feng, S. An Agent-Based Geographical Information System. *Knowl. Based Syst.* **2001**, *14*, 233–242.
22. Dreżewski, R.; Doroz, K. An Agent-Based Co-Evolutionary Multi-Objective Algorithm for Portfolio Optimization. *Symmetry* **2017**, *9*, 168.
23. Wang, D.; Ren, H.; Shao, F. Distributed Newton Methods for Strictly Convex Consensus Optimization Problems in Multi-Agent Networks. *Symmetry* **2017**, *9*, 163.
24. West, D.; Dellana, S. Diversity of ability and cognitive style for group decision processes. *Inf. Sci.* **2009**, *179*, 542–558.
25. Zamfirescu, C.B.; Duta, L.; Iantovics, L.B. On investigating the cognitive complexity of designing the group decision process. *Stud. Inform. Control* **2010**, *19*, 263–270.
26. Iantovics, L.B.; Rotar, C. A Novel Metric for Comparing the Intelligence of Two Swarm Multiagent Systems. *J. Artif. Intell.* **2016**, *9*, 39–44.
27. Besold, T.; Hernández-Orallo, J.; Schmid, U. Can Machine Intelligence be Measured in the Same Way as Human intelligence? *Künstl. Intell.* **2015**, *29*, 291–297.
28. Kannan, B.; Parker, L.E. Metrics for quantifying system performance in intelligent, fault-tolerant multi-robot teams. In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 951–958.
29. Park, H.J.; Kim, B.K.; Lim, K.Y. Measuring the machine intelligence quotient (MIQ) of human-machine cooperative systems. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **2001**, *31*, 89–96.

30. Schreiner, K. Measuring IS: Toward a US standard. *IEEE Intell. Syst. Their Appl.* **2000**, *15*, 19–21.
31. Arik, S.; Iantovics, L.B.; Szilagyi, S.M. OutIntSys—A Novel Method for the Detection of the Most Intelligent Cooperative Multiagent Systems. In Proceedings of the 24th International Conference on Neural Information Processing, Guangzhou, China, 14–18 November 2017; Volume 10637, pp. 31–40.
32. Wallace, C.S.; Dowe, D.L. Minimum message length and Kolmogorov complexity. *Comput. J.* **1999**, *42*, 270–283.
33. Wallace, C.S. *Statistical and Inductive Inference by Minimum Message Length*; Springer: Berlin/Heidelberg, Germany, 2005.
34. Dowe, D.L. MML, hybrid Bayesian network graphical models, statistical consistency, invariance and uniqueness. In *Handbook of the Philosophy of Science*; Bandyopadhyay, P.S., Forster, M.R., Eds.; Elsevier: Amsterdam, The Netherlands, 2011; Volume 7, pp.901–982.
35. Dowe, D.L.; Hajek, A.R. A computational extension to the Turing Test. In Proceedings of the 4th Conference of the Australasian Cognitive Science Society, Melbourne, Australia, 28 November–1 December 2013.
36. Dowe, D.L.; Hajek, A.R. A non-behavioural, computational extension to the Turing Test. In Proceedings of the International Conference on Computational Intelligence and Multimedia Application, Gippsland, Australia, 7–10 February 1998; pp. 101–106.
37. Wallace, C.S.; Boulton, D.M. An information measure for classification. *Comput. J.* **1968**, *11*, 185–194.
38. Sterret, S.G. Turing on the Integration of Human and Machine Intelligence. In *Philosophical Explorations of the Legacy of Alan Turing*; Floyd, J., Bokulich, A., Eds.; Springer: Cham, Switzerland, 2017; Volume 324, pp. 323–338.
39. Ferrucci, D.; Levas, A.; Bagchi, S.; Gondek, D.; Mueller, E.T. Watson: Beyond Jeopardy! *Artificial Intelligence* **2013**, *199*, 93–105.
40. Legg, S.; Hutter, M. A Formal Measure of Machine Intelligence.In Proceedings of the 15th Annual Machine Learning Conference of Belgium and The Netherlands (Benelearn 2006), Ghent, Belgium, 11–12 May 2006; pp. 73–80.
41. Hibbard, B. Measuring Agent Intelligence via Hierarchies of Environments. In *Artificial General Intelligence AGI 2011*; Schmidhuber, J., Thórisson, K.R., Looks, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6830, pp. 303–308.
42. Anthon, A.; Jannett, T.C. Measuring machine intelligence of an agent-based distributed sensor network system. In *Advances and Innovations in Systems, Computing Sciences and Software Engineering*; Elleithy, K., Ed.; Springer: Berlin/Heidelberg, Germany, 2007; pp. 531–535.
43. Hernandez-Orallo, J.; Dowe, D.L. Measuring universal intelligence: Towards an anytime intelligence test. *Artif. Intell.* **2010**, *174*, 1508–1539.
44. Iantovics, L.B.; Emmert-Streib, F.; Arik, S. MetrIntMeas a novel metric for measuring the intelligence of a swarm of cooperating agents. *Cogn. Syst. Res.* **2017**, *45*, 17–29.
45. Winklerova, Z. Maturity of the Particle Swarm as a Metric for Measuring the Collective Intelligence of the Swarm. In *Advances in Swarm Intelligence, ICSI 2013*; Tan, Y., Shi, Y., Mo, H., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7928, pp. 40–54.
46. Iantovics, L.B.; Rotar, C.; Niazi, M.A. MetrIntPair—A Novel Accurate Metric for the Comparison of Two Cooperative Multiagent Systems Intelligence Based on Paired Intelligence Measurements. *Int. J. Intelli. Syst.* **2018**, *33*, 463–486.
47. Liu, F.; Shi, Y.; Liu, Y. Intelligence quotient and intelligence grade of artificial intelligence. *Ann. Data Sci.* **2017**, *4*, 179–191.
48. Detterman, D.K. A challenge to Watson. *Intelligence* **2011**, *39*, 77–78.
49. Sanghi, P.; Dowe, D.L. A computer program capable of passing I.Q. tests. In Proceedings of the Joint International Conference on Cognitive Science, 4th ICCS International Conference on Cognitive Science and 7th ASCS Australasian Society for Cognitive Science (ICCS/ASCS-2003) Sydney, Australia, 13–17 July 2003; pp. 570–575.
50. Campbell, M.; Hoane, A.J.; Hsu, F. Deep Blue. *Artif. Intell.* **2002**, *134*, 57–83.
51. Iantovics, L.B.; Rotar, C.; Nechita, E. A novel robust metric for comparing the intelligence of two cooperative multiagent systems. *Procedia Comput. Sci.* **2016**, *96*, 637–644.
52. Chakravarti, I.M.; Laha, R.G.; Roy, J. *Handbook of Methods of Applied Statistics*; John Wiley and Sons: Hoboken, NJ, USA, 1967; Volume I, pp. 392–394.

53. Lilliefors, H. On the Kolmogorov–Smirnov test for normality with mean and variance unknown. *J. Am. Stat. Assoc.* **1967**, *62*, 399–402.
54. Lilliefors, H. On the Kolmogorov–Smirnov test for the exponential distribution with mean unknown. *J. Am. Stat. Assoc.* **1969**, *64*, 387–389.
55. Dallal, G.E.; Wilkinson, L. An analytic approximation to the distribution of Lilliefors's test statistic for normality. *Am. Stat.* **1986**, *40*, 294–296.
56. Markowski, C.A.; Markowski, E.P. Conditions for the Effectiveness of a Preliminary Test of Variance. *Am. Stat.* **1990**, *44*, 322–326.
57. Bartlett, M.S. Properties of sufficiency and statistical tests. *Proc. R. Soc. Lond. A* **1937**, *160*, 268–282.
58. Snedecor, G.W.; Cochran, W.G. *Statistical Methods*, 8th ed.; Iowa State University Press: Iowa, IA, USA, 1989.
59. Ross, S.M. Peirce's Criterion for the Elimination of Suspect Experimental Data. *J. Eng. Technol.* **2003**, *2*, 1–12.
60. Zerbet, A.; Nikulin, M. A new statistics for detecting outliers in exponential case. *Commun. Stat. Theory Methods* **2003**, *32*, 573–583.
61. Stigler, S.M. Mathematical statistics in the early states. *Ann. Stat.* **1978**, *6*, 239–265.
62. Dean, R.B.; Dixon, W.J. Simplified Statistics for Small Numbers of Observations. *Anal. Chem.* **1951**, *23*, 636–638.
63. Barnett, V.; Lewis, T. Evolution by gene duplication. In *Outliers in Statistical Data*, 3rd ed.; Wiley: Hoboken, NJ, USA, 1994.
64. Motulsky, H. GraphPad InStat Version 3. In *The InStat Guide to Choosing and Interpreting Statistical Tests*; GraphPad Software, Inc.: La Jolla, CA, USA, 2003.
65. Fisher, R.A. On the "Probable Error" of a Coefficient of Correlation Deduced from a Small Sample. *Metron* **1921**, *1*, 3–32.
66. Kruskal, W.H.; Wallis, W.A. Use of ranks in one-criterion variance analysis. *J. Am. Stat. Assoc.* **1952**, *47*, 583–621.
67. Dunn, O.J. Multiple comparisons using rank sums. *Technometrics* **1964**, *6*, 241–252.
68. Tukey, J. Comparing Individual Means in the Analysis of Variance. *Biometrics* **1949**, *5*, 99–114.
69. Sosnoff, J.J.; Heffernan, K.S.; Jae, S.Y.; Fernhall, B. Aging, hypertension and physiological tremor: The contribution of the cardioballistic impulse to tremorgenesis in older adults. *J. Neurol. Sci.* **2013**, *326*, 68–74.
70. Lee, J.Y.; Shin, S.Y.; Park, T.H.; Zhang, B.T. Solving traveling salesman problems with DNA molecules encoding numerical values. *Biosystems* **2004**, *78*, 39–47.
71. Fischer, A.; Fischer, F.; Jager, G.; Keilwagen, J.; Molitor, P.; Grosse, I. Computational Recognition of RNA Splice Sites by Exact Algorithms for the Quadratic Traveling Salesman Problem. *Computation* **2015**, *3*, 285–298.
72. Kim, E., Lee, M., Gatton, T.M., Lee, J., Zang, Y. An Evolution Based Biosensor Receptor DNA Sequence Generation Algorithm. *Sensors* **2010**, *10*, 330–341.
73. Boctor, F.F.; Laporte, G.; Renaud, J. Heuristics for the traveling purchaser problem. *Comput. Oper. Res.* **2003**, *30*, 491–504.
74. Dantzig, G.B.; Ramser, J.H. The Truck Dispatching Problem. *Manag. Sci.* **1959**, *6*, 80–91.
75. Dantzig, G.; Fulkerson, D.; Johnson, S. Solution of a large scale traveling salesman problem. *Oper. Res.* **1954**, *2*, 393–410.
76. Ouaarab, A.; Ahiod, B.; Yang, X.S. Discrete cuckoo search algorithm for the travelling salesman problem, *Neural Comput. Appl.* **2014**, *24*, 1659.
77. Huang, Z.G.; Wang, L.G.; Xu, Z.; Cui, J.J. An efficient two-step iterative method for solving a class of complex symmetric linear systems. *Comput. Math. Appl.* **2018**, in press.
78. Koczy, L.T.; Foldesi, P.; Tuu-Szabo, B. Enhanced discrete bacterial memetic evolutionary algorithm—An efficacious metaheuristic for the traveling salesman optimization. *Inf. Sci.* **2017**, in press.
79. Jonker, R.; Volgenant, T. Transforming asymmetric into symmetric traveling salesman problems. *Oper. Res. Lett.* **1983**, *2*, 161–163.
80. Arthanari, T.S.; Usha, M. An alternate formulation of the symmetric traveling salesman problem and its properties. *Discret. Appl. Math.* **2000**, *98*, 173–190.
81. Smith, T.H.C.; Meyer, T.W.S.; Thompson, G.L. Lower bounds for the symmetric travelling salesman problem from Lagrangean relaxations. *Discret. Appl. Math.* **1990**, *26*, 209–217.
82. Holldobler, B.; Wilson, E.O. *The Ants*; Harvard University Press: Cambridge, MA, USA, 1990.

83. Higashi, S.; Yamauchi, K. Influence of a Supercolonial Ant Formica(Formica) yessensis Forel on the Distribution of Other Ants in Ishikari Coast. *Jpn. J. Ecol.* **1979**, *29*, 257–264.
84. Giraud, T.; Pedersen, J.S.; Keller, L. Evolution of supercolonies: The Argentine ants of southern Europe. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 6075–6079.
85. Sim, Y.B.; Lee, S.G.; Lee, S. Function-Oriented Networking and On-Demand Routing System in Network Using Ant Colony Optimization Algorithm. *Symmetry* **2017**, *9*, 272.
86. Crisan, G.C.; Pintea, C.M.; Palade, V. Emergency Management Using Geographic Information Systems: Application to the first Romanian Traveling Salesman Problem Instance. *Knowl. Inf. Syst.* **2017**, *50*, 265–285.
87. Everitt, B. *The Cambridge Dictionary of Statistics*; Cambridge University Press: New York, NY, USA, 1998.
88. Marusteri, M.; Bacarea, V. Comparing groups for statistical differences: How to choose the right statistical test? *Biochem. Med.* **2010**, *20*, 15–32.
89. Mann, H.B.; Whitney, D.R. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *Ann. Math. Stat.* **1947**, *18*, 50–60.
90. Fay, M.P.; Proschan, M.A. Wilcoxon–Mann–Whitney or *t*-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. *Stat. Surv.* **2010**, *4*, 1–39.
91. McDonald, J.H. *Handbook of Biological Statistics*, 3rd ed.; Sparky House Publishing: Baltimore, MD, USA, 2014.
92. Fagerland, M.W.; Sandvik, L. The Wilcoxon-Mann-Whitney Test under scrutiny. *Stat. Med.* **2009**, *28*, 1487–1497.
93. Slavin, R. *Educational Psychology Theory and Practice*, 9th ed.; Allyn and Bacon: Boston, MA, USA, 2009; p. 117.
94. Gardner, H. *Frames of Mind: The Theory of Multiple Intelligences*, 3rd ed.; Basic Books: New York, NY, USA, 2011.
95. Visser, B.A.; Ashton, M.C.; Vernon, P.A. g and the measurement of Multiple Intelligences: A response to Gardner. *Intelligence* **2006**, *34*, 507–510.
96. Bell, E.A.; Boehnke, P.; Harrison, T.M.; Mao, W.L. Potentially biogenic carbon preserved in a 4.1 billion year-old zircon. *Proc. Natl. Acad. Sci. USA* **2015**, *112*, 14518–14521.


symmetry

MDPI

Article

# IntraClusTSP—An Incremental Intra-Cluster Refinement Heuristic Algorithm for Symmetric Travelling Salesman Problem

**László Kovács [1], László Barna Iantovics [2,*] and Dimitris K. Iakovidis [3]**

[1] Department of Information Technology, University of Miskolc, H-3515 Miskolc-Egyetemváros, Hungary; kovacs@iit.uni-miskolc.hu
[2] Department of Informatics, University of Medicine, Pharmacy, Sciences and Technology of Targu Mures, R-540139 Târgu Mureș, Romania
[3] Department of Computer Science and Biomedical Informatics, University of Thessaly, GR-35131 Lamia, Greece; dimitris.iakovidis@ieee.org
* Correspondence: ibarna@science.upm.ro

Received: 16 October 2018; Accepted: 13 November 2018; Published: 22 November 2018


**Abstract:** The Symmetric Traveling Salesman Problem (sTSP) is an intensively studied NP-hard problem. It has many important real-life applications such as logistics, planning, manufacturing of microchips and DNA sequencing. In this paper we propose a cluster level incremental tour construction method called Intra-cluster Refinement Heuristic (IntraClusTSP). The proposed method can be used both to extend the tour with a new node and to improve the existing tour. The refinement step generates a local optimal tour for a cluster of neighbouring nodes and this local optimal tour is then merged into the global optimal tour. Based on the performed evaluation tests the proposed IntraClusTSP method provides an efficient incremental tour generation and it can improve the tour efficiency for every tested state-of-the-art methods including the most efficient Chained Lin-Kernighan refinement algorithm. As an application example, we apply IntraClusTSP to automatically determine the optimal number of clusters in a cluster analysis problem. The standard methods like Silhouette index, Elbow method or Gap statistic method, to estimate the number of clusters support only partitional (single level) clustering, while in many application areas, the hierarchical (multi-level) clustering provides a better clustering model. Our proposed method can discover hierarchical clustering structure and provides an outstanding performance both in accuracy and execution time.

**Keywords:** Symmetric Traveling Salesman Problem; symmetry; symmetric distance matrix; Nearest Neighbour method; Chained Lin-Kernighan refinement algorithm; Intra-cluster Refinement; clustering; optimal number of clusters; similar elements

## 1. Introduction

Symmetric graphs have many real-life applications as vehicle routing, warehouse logistics, planning circuit boards, virtual networking [1–4]. The goal of the base Symmetric Traveling Salesman Problem (sTSP) is to find the shortest Hamiltonian cycle in a graph. The Hamiltonian cycle visits each vertex exactly once. The length of the path is calculated with the sum of the corresponding edge weights. The weight values of the graph edges, in general case, are given with a squared matrix of non-negative values. In this paper, we are focusing on Euclidean TSP (eTSP) problems where the graph nodes correspond to points in the Euclidean metric space and the weights are equal to the Euclidean distances between these points. In this case we get a symmetric distance matrix. The generation of

the shortest Hamiltonian cycle is an NP-hard problem [5] usually formulated as an integer linear programming problem [6].

Regarding the complexity of the sTSP, two different problems are usually investigated: the Decision problem (DTSP) and the Optimization problem (OTSP). DTSP aims to determine whether a Hamiltonian cycle of length not greater than a given value exists. The goal of OTSP is to find the Hamiltonian cycle of minimal length.

In the brute-force solution method, all possible permutations of the nodes are tested to select the optimal route. This approach requires $O(N!)$ execution cost. Regarding the other alternative exact solution methods, we can emphasize the integer linear programming approach and the Held-Karp algorithm. In the case of integer linear programming formulation [7], we have $O(N^2)$ variables and subtour elimination constraints. As a direct solution is unfeasible, the model is relaxed to find a solution. In the proposal [8] a novel representation form was introduced to reduce the number of subtour elimination constraints. In Reference [9], the related integer linear programming problem is based on the two-commodity network flow formulation of the TSP.

The Held-Karp method [10] uses the dynamic programming approach. The algorithm follows the idea of successive approximation; the global problem is decomposed into a set of related subproblems and the optimal solutions of the subproblems are composed into an optimal global solution. Although, this method provides a better time cost efficiency than the brute-force algorithm, it belongs to exponential complexity class, having a worst-case cost $O(2^N N^2)$. Another drawback of the method is that it raises a significant space requirement too.

In Reference [11] the branch-and-bound approach was implemented to determine the exact optimal tour. The method builds up a state-tree that manages the paths already processed. Each node stores a path description together with its cost values involving a lower bound cost value too. The construction of the state tree requires $O(N!)$ costs in the worst case. In the [12] a genetic node is assigned to an assignment problem. The related subtours are broken by creating subproblems in which all edges of the subtour are prohibited.

Due to high computational costs of the exact solution methods, the heuristic optimization of sTSP is one of the most widely investigated combinatorial optimization problem. One group of the heuristic methods use algorithms for direct route construction. As the main goal is to minimize the sum of a fixed number of edge weights, a sound heuristic is to minimize the components in the sum. Thus, the constructional heuristic methods are in general aimed for selecting the edges of minimal length. In the case of Nearest Neighbour method [13], the algorithm starts with a random selection of a vertex. In each iteration step, the nearest free vertex is selected and it will be connected to the current node. In the Greedy Edge Insertion heuristic method [14], the edges are ordered by their weight values. The algorithm inserts the shortest available edge into the route in every iteration step. During the construction process there are two constraints to be met: (a) any node is connected to exactly two other nodes; (b) no cycle can exist with less than N edges. The Greedy Vertex Insertion algorithm [15] extends the existing route with a vertex having the lowest cost increase. A similar approach was implemented in the Boruvka algorithm [16] where the edges are processed in length order. An edge with minimal length is inserted into the tour if it does not affect the integrity of the current route.

The Fast Recursive Partitioning method [17] performs a hierarchical clustering of the nodes corresponding to points in the Euclidean space. The points are structured into a hierarchical tree, similarly to the R-tree structure. The leaf nodes contain a smaller number of points, with a given capacity. In first phase, the algorithm performs a TSP route generation for each of the leaf buckets and in the second phase, the local routes are merged into a global tour. The Karp's Partitioning Heuristic [18] is based on a similar hierarchical decomposition technique but it uses a more sophisticated patch-based method. The Double Minimum Spanning Tree algorithm [19] and its improved version, the Christofides Algorithm [20], generates first a minimal spanning tree for the input graph and adjusts this tree with a minimum-weight matching.

Another family of heuristic methods aims at the improvement of existing solutions found so far. Having one or more initial suggestions, the algorithm tries to find a better solution. The improvement heuristics can achieve significantly better results than the direct construction methods [6]. These algorithms use iterations and random search components; thus the execution time is here usually higher. The k-opt method [21] belongs to the family of local search heuristics. Similarly to the string edit distance, a tour distance is defined between two edge sequences. The tour t' is considered in the k-neighbourhood of t if the distance between t and t' is not greater than $k$. Having an initial tour, this method repeatedly replaces the current tour with a tour in its neighbourhood providing a smaller tour length. Due to the higher time cost of the neighbourhood construction process, the variants with lower k values (2-opt and 3-opt) are preferred in the practical implementations.

The Lin-Kernighan [22] algorithm improves the efficiency of the standard k-opt approaches by using a flexible neighbourhood construction. The method first determines the optimal $k$ value for the k-opt method and performs the search operation in the dynamic k-neighbourhood environment, where each move is composed of special 2-opt and 3-opt moves. As the quality of the base Lin-Kernighan local optimization methods depends significantly on the quality of the initial route, a usual approach is to repeat the local search with different initial routes and return the best result. The approach presented by [23] uses a different idea to work harder on the current tours using kicks on the tour found by Lin-Kernighan. The resulting algorithm is known as Chained Lin-Kernighan method.

There are also approaches using generic Evolutionary Optimization [24] methods for the shortest tour problem. Such a method generates a set of initial tours and using the route length as a fitness function, the next generation is produced with the application of the reproduction, crossover and mutation operators. As the mutation and crossover has a random nature, the quality of the result is weak for large search space problems. In Reference [25], the predictability of TSP optimization is investigated analysing different bacterial evolutionary algorithms with local search.

In the Multi-level Approach [26], the tour is constructed with a hierarchy of increasingly coarse approximations. Having an initial problem on $N$ nodes, the algorithm first fixes an edge at every level, thus the next level optimizes a problem of smaller size having only $(N - 1)$ nodes. After generating an initial tour, a usual refinement phase is executed to improve the tour quality. The Tour-merging Method [27] is based on the observation that the routes of good quality usually share a large set of common edges. The algorithm uses specific heuristics to generate near-optimal tours and dynamic programming to unify the partial solutions into a common solution.

There is a rich literature on detailed analysis and performance comparison of the main heuristic methods (Nearest Neighbour, Nearest Insertion, Tabu Search, Lin-Kernighan, Greedy, Boruvka, Savings and Genetic Algorithm). Based on the results presented in Reference [16,19,28]: (a) the fastest algorithms are the Greedy and Savings method but they provide an average tour quality; (b) the Nearest Neighbour and Nearest Insertion algorithms are dominated by the Greedy and Savings methods both in time and tour quality factors; (c) the best route quality can be achieved by the application of 3-opt/5-opt methods (Lin-Kernighan and Helsgaun); (d) considering both the time and tour quality, the Chained Lin-Kernighan algorithm proves the best performance; (e) the Evolutionary and Swarm optimization methods are dominated by the k-opt methods both in time and tour quality factors; (f) the Tour-merging methods applied on the Chained Lin-Kernighan algorithm can improve the tour quality at some level but it requires a significantly higher time cost.

Most of the available heuristic methods work in a batch mode, where the full graph is presented as input. In this case the algorithm can get all information already at the start of the tour generation. An alternative approach is the incremental mode where the graph is initially empty and it is extended with new nodes incrementally. This can happen in some application areas as knowledge engineering or transportation problems where new concepts/locations can be added to the existing network. Having only a batch algorithm, after the insertion of a new node, we should rerun the full optimization process to get the new optimal tour. In these cases the incremental algorithms can provide a better solution than the standard batch methods.

Considering the main heuristic methods, only few can be adapted to the incremental construction approach. In the family of constructional heuristic methods, the methods usually process the edges in some selected order. For example, in Greedy Edge Insertion heuristic method [14] or in Boruvka algorithm [16], the edges are sorted by the length value. Greedy Vertex Insertion algorithm [15] extends the existing route with a vertex having the lowest cost increase. In the case of Nearest Neighbour method [13], the shortest free edge is selected related to the actual node. In all cases, if we extend the graph with a random new element and process this element with the mentioned methods, the resulted route will be usually suboptimal. In a batch mode, this element would be processed in an earlier step. The heuristics methods form the other groups use such operations (improvements, hierarchical decomposition) which are defined on the complete graph or on a complete subgraph.

In this paper, we propose a novel algorithm, called IntraClusTSP that can be used in Euclidean TSP for both incremental tour construction and tour refinement. The method first selects one or more clusters in the object graph and then it performs cluster-level route optimization for each cluster. In the final step, the optimized local routes are merged with the current global route yielding a new optimal route. This approach is based on the observation that adjacent nodes in the shortest Hamiltonian cycle in eTSP are usually nearest neighbour nodes. The IntraClusTSP refinement method provides a general framework for route optimization as it can use any current TSP methods to perform cluster level optimization. In our model, we use Chained Lin-Kernighan method for local optimization. The clusters selected for refinement may be overlapping clusters. Based on the performed tests, the proposed method provides superior efficiency for incremental route construction.

The rest of this paper is structured as follows. In the next section, a survey on TSP heuristics using incremental or cluster-based optimization is presented. Section 3 presents the motivation for the development of the proposed IntraClusTSP method, the formal model and the constructed algorithms. Section 4 focuses on its cost model and cost analysis. It presents the test results of the empirical efficiency comparison of the proposed method with the several TSP solution algorithms. Section 5 demonstrates the application of IntraClusTSP algorithm in solving of a data analysis problem.

## 2. Incremental and Segmentation-Based Approaches in Solving eTSP

In the TSP terminology, the term incremental insertion heuristic refers to methods where the optimal tour is constructed by extension steps where in each step, the route is extended with a single node. The most widely known methods of this group are Nearest point insertion [29], Furthest point insertion [30] or Random point insertion [31]. Based on the literature, the Furthest point insertion provides the best tour length. In this case, the point $x$ as the solution of

$$argmax_{x \in T''} \left\{ min_i^T (d(x_i, x) + d(x_{i+1}, x)) \right\},$$

is selected to be inserted. In the formula, $T$ denotes the current route and $T''$ is its complement. In our problem domain, this method is not suitable, as in every insertion cycle, the $T''$ set contains only one node, the rest nodes are not known yet. Similarly, the Nearest point insertion method selects the nearest node from the points not linked into the tour yet. Thus, only the Random insertion heuristic can be used for our problem domain.

Considering the approximation efficiency of the insertion algorithms, Rosenkrantz et al. [32] has been proven that every insertion algorithm provides an approximation threshold $O(log(N))$. The Furthest insertion method that performs better than the other method has a constant lower bound [33] of 2.43 for eTSP problems. Regarding the efficiency of the Random insertion algorithm, Azar has proven in Reference [31] that the worst case approximation factor can be given with

$$\Omega(\frac{log(log(N)}{log(log(log(N)))})$$

The shape of the worst case factor function is given in Figure 1.

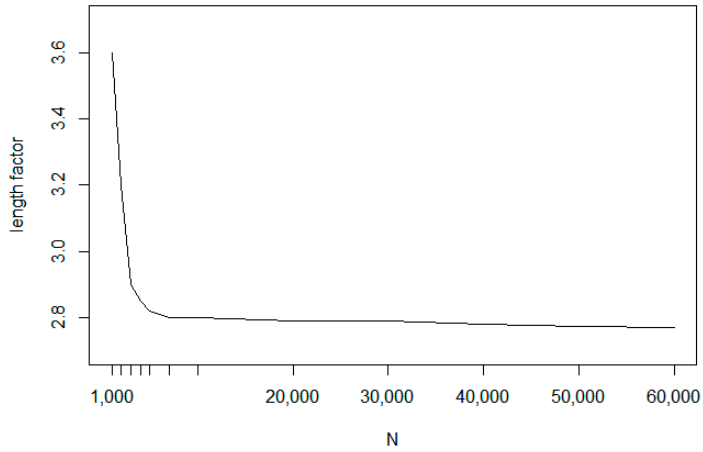



**Figure 1.** Worst case approximation bound function.

As this theorem proves the Random insertion method provides in general about 15% weaker result than the Furthest insertion method. In the Random insertion approach, the position of the new node within the route is calculated with a local optimization step. One option is to minimize the tour length increase:

$$min_i^T(d(x_i, x) + d(x_{i+1}, x))$$

Another option is to connect the new node to the closest tour element:

$$min_i^T(d(x_i, x))$$

and take the neighbour node with shortest distance as the second adjusted node.

One of the first publications on incremental tour generation [34] is made by T.M. Cronin in 1990. The algorithm ensures optimality as each city is inserted. The author has developed a dynamic programming algorithm which begins with a baseline tour consisting of the outer convex hull of cities and proceeds by adding a city at a time to the interior.

The proposed method is based on the following theoretical result: the shortest tour containing $k$ cities is a quartic and hyperbolic function of the shortest tour containing $(k-1)$ cities. It can be proven that an optimal tour must preserve the order defined on the convex hull of nodes [35,36]. In this model, a perturbation is a sub-tour which leads into the interior of the hull through two adjacent hull vertices, to capture nodes which do not lie on the hull. Considering the insertion a new node into the tour, the tour is extended by inserting the new node between those two nodes for which the distance is smallest. The tests were executed on small examples containing only 127 nodes.

A generalization of the insertion method is presented in Reference [37] where during the insertion procedure the two neighbouring nodes of the new item are not necessarily consecutive.

As the practical experiences show [19] the most efficient methods use a mixed approach where a refinement phase is applied on the tour constructed initially. In our investigation, we focus on segment-level refinement optimization. The motivation on segmentation in eTSP is based on the experience that the optimal route usually connects near vertices in the plane. The segmentation generates a set of smaller optimization subproblems to be solved.

This TSP domain was introduced in 1975 by the research paper [38]. In CTSP (Clustered Traveling Salesman Problem), the salesman must not only visit each city once and only once but a subset (cluster) of these cities must be visited contiguously. The presented method first reduces the weights of every intra-cluster edges. Then, a standard branch and bound optimization is applied to the whole graph. The performed weight reduction ensures that the optimization algorithm will generate the required intra-cluster routes. Later, several new methods were proposed to solve the CTSP problem, like the Langrangian method using spanning tree constructions for the graph optimization [39].

The cluster-based segmentation can be considered as an integrity constraints but it can be used a tool for reduce the execution costs of the optimization algorithms. The segmentation as a divide and conquer approach was introduced among others in Reference [40], where the algorithm starts with the segmentation of the nodes into disjoint clusters using a K-means clustering algorithm. In the next phase of the proposed method, the local clusters are optimized using the Lin-Kernighan method yielding a set of optimal local tours. The local tours are merged into a global tour in the final step. In the merge phase, the cluster centroids are calculated first, then $k$-nearest elements in the global tour are determined. Next, each of the $k$ nodes will be tested to determine the cost of inserting the cluster tour in place of the following edge. The given local tour will be inserted before the node with best cost value.

The idea of combining local clustering (segmentation) in generation of initial route is used in many current proposals. In the literature, we can find many variants of this segmentation approach, as Geometric Partitioning [41], Tour-Based Partitioning [42] and Karp's Partitioning Heuristic (Karp) [43]. In more complex cases, like [44], the method constructs a hierarchy of segmentations in order to provide a cluster leaves with small amount of graph nodes. In Reference [17], the graph nodes are separated into four disjoint groups corresponding to the different sides of a rectangle. In Reference [45], the route generated by merging the cluster level clusters is refined with a genetic algorithm heuristic. In Reference [46], the genetic algorithm and the ant colony optimization are used to find the optimal local path for the clusters. In the final step, a simple method for choosing clusters and nodes is presented to connect all clusters in the TSP. The Tour-merging Method [27,35] is based on the observation that the routes of good quality usually share a large set of common edges. The algorithm uses specific heuristics to generate near optimal tours and specific dynamic programming techniques to unify the partial solutions into a common solution.

The extensive literature survey that we made shows that the targeted incremental graph and optimal route construction approach attracted little attention and no detailed analysis can be found. In contrary to the rich variety of optimization algorithms on general TSP, only the Random point insertion method can be used directly as an incremental TSP method. As the Random point insertion method is considered as a sub-optimal algorithm dominated by many other methods (like Furthest point insertion, Chained Lin-Kernighan), our motivation is to propose a novel incremental method having a better optimization efficiency than Random point insertion method has and having a better execution cost than the standard non-incremental TSP methods have.

## 3. IntraClusTSP: Intra-Cluster Refinement Method for Incremental sTSP

### *3.1. Motivations for the Development of a Novel Algorithm*

Although, a variety of methods for the tour improvement heuristics have been proposed [36,47], the best methods are based on the following two main optimization approaches: (a) systematic exchange of edge tuples (the k-opt optimization methods use this approach); and, (b) hierarchical decomposition of the original problem into smaller problems.

Due to the complexity of the k-opt method, usually only a lower k value is used in optimization process. One refinement step relocates the current status vector to a neighbouring position, modifying arbitrary segments of the tour. In this sense, the k-opt method aims at a global optimization, there is no way to perform the optimization only on a predefined segment of the node set.

The main goal of our proposed method is to provide a different approach to the tour optimization that can focus on a cluster of edges in the tour reordering process. The algorithm is based on the idea of 'divide and conquer' concept, that is, it selects an arbitrary segment (a cluster of nodes with the corresponding edges among them) and then an efficient known TSP algorithm is used to solve this sub-problem. In the second phase of the iteration, the generated local tour is merged with the rest (unchanged) part of the initial tour. The proposed approach reuses some known concepts but it differs from the existing approaches in many aspects:

- The proposed method performs a tour improvement and not an initial tour construction (unlike the Greedy, Nearest Neighbour or Savings methods).
- The segments may be overlapping; the same segment can be processed several times.
- The proposed model can be used as an incremental TSP construction method. The new item is inserted into the current route using the simple Random point insertion method and then a cluster level refinement phase is executed.
- The link between a segment and the main route is very flexible, there may be any number of connection edges
- It differs from the k-opt, Lin-Kernighan and Helsgaun methods because the size of the sub-graph to be improved may be arbitrary and the algorithm of the subgraph optimization can be considered as a black box optimization.
- Unlike the multi-level approach, this method performs a single level edge fixing and not a hierarchic refinement.
- Unlike the tour-merging method, the method generates only a sub-tour to be replaced in the original tour.

### *3.2. Formal Model and Algorithm of the Proposed Method*

Let $V = \{v_1, v_2, \ldots, v_N\}$ denote the set of $N$ positions in a metric space with a distance function

$$d : V \times V \to \Re$$

The corresponding distance matrix is denoted by $D_V$ A Hamiltonian cycle of $V$ visiting each position only once is denoted by $t_V$. The permutation related to Hamiltonian cycle t is given by $\pi^t$. The tour length of $t$ is defined as

$$d(t) = \sum_{i=1}^{N-1} d\left(\pi_{i+1}^t, \pi_i^t\right)$$

where $\pi_i^t$ denotes the $i$-th element of the permutation. The set of all Hamiltonian cycles is given with

$$T_V = \{t_V\}$$

Having an initial approximation $t_V^0$ of the optimal Hamiltonian cycle for $V$, the intra-cluster reordering tour improvement method selects a subset $V' \subseteq V$ and using an appropriate $D_{V'}$ distance matrix, a local optimal tour ${t'}_{V'}$ is generated. The distance matrix $D_{V'}$ is constructed from $D_V$ on the following way. First, we decompose $V'$ into two parts: a set of internal nodes and a set of border nodes. A node is a border node if one of its adjacent nodes is an element of $V'$ and the other adjacent node in not element of $V'$. Two border nodes are linked nodes if there exists a route in $V \backslash V'$ connecting these elements. The distance values are given with

$$D_{V'}[i,j] = \begin{cases} D[i,j], & \text{if i or j denotes an internal node} \\ 0, & \text{if i and j are linked border nodes} \\ \infty, & \text{if i and j are not linked border nodes} \end{cases}$$

If $d(t_{V'}^0) > d({t'}_{V'})$, then the new route is involved into the global tour. In the optimization phase, two linked border nodes are merged into a single virtual border node. Thus the tour sections in $V \backslash V'$ remain hidden is this phase. It can be seen that the distances related to the virtual border nodes are not symmetric as these nodes have two physical positions.

In the merge phase, the generated cluster level route which contains both the internal and border nodes is merged into the global tour. In this phase, the tour sections hidden in the border nodes are uncovered again. Using a tour merge operation, the improved route is given with

$$t_V^{i+1} \;=\; (t_V^i \backslash t_{V'}^i) \cup t'_{V'}$$

Thus, the route sequence $t_V^1, t_V^2, \ldots, t_V^m$ is an improvement sequence of the initial route $t_V^0$.

As an example, let us take the tour given in Figure 2a. In the example, 17 nodes are given. The internal nodes (modes in the cluster) are given in white colour. There are 6 border nodes given in grey and there are 5 external nodes in black. For the local optimization phase, three virtual border nodes are generated, the related pairs are denoted with thick edges. The cluster-level local optimization process involves 10 nodes. The generated local optimal tour is shown in Figure 2b. After uncovering merging with the external, hidden section, we get the new, refined global tour (Figure 2c).

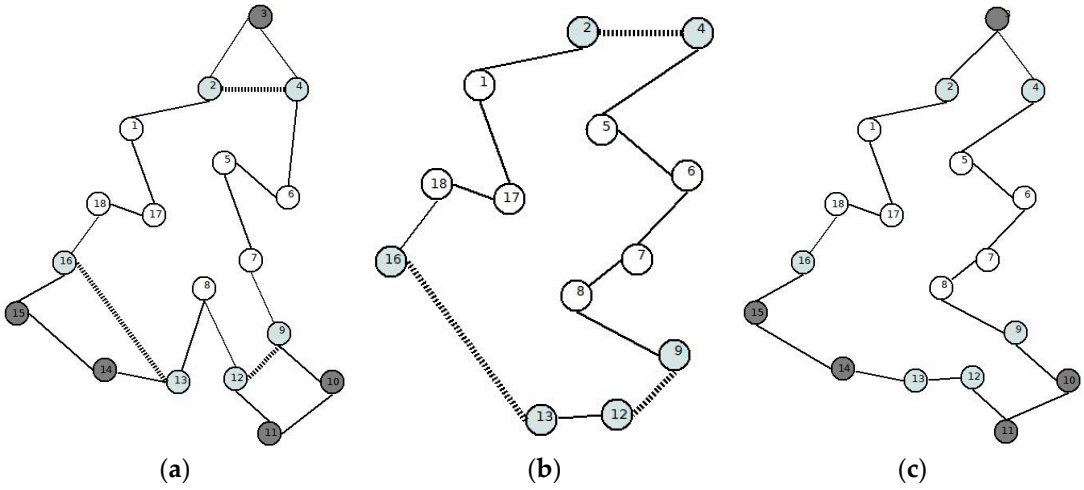



**Figure 2.** (**a**–**c**) Sample cluster level tour optimization.

If the IntraClusTSP is used to general tour refinement, then we apply the Quality Threshold Clustering (QTC) to provide an appropriate segmentation. This method uses a partitioning clustering [48] proposed for analysis of co-expressed genes, generating clusters with a bounded intra-cluster diameter.

If the IntraClusTSP is used to add a new node in the incremental route construction task, then we should select a cluster where the new node is near to the cluster centre. We applied a minimum tour extension concept, where

$$min_i^T(d(x_i, x) + d(x_{i+1}, x))$$

is met. After this insertion phase, we determine a cluster around the new element and perform a cluster level refinement with the IntraClusTSP method.

An important motivation was in our investigation the fact that the search space of larger TSP problems are significantly more complex than the search space of smaller problems. A deep analysis of the corresponding search landscape can be found in Reference [49] where a statistical model was introduced to estimate the number of local optima in the search space for different problems sizes. In the proposed method, random sampling technology is used to determine the corresponding model parameters. A similar analysis was presented later in Reference [50] for a larger problem domain. An important empirical result of the investigation on Euclidean TSP problems was that the number of local optima ($C_{lopt}$) grows exponentially in the function of the problem size ($N$):

$$C_{lopt}(N) \in O(e^{Nlog(N)})$$

To demonstrate the complexity of the optimization landscape, we present an experiment in a low complexity permutation space. Figure 1a,b shows the space of all permutations ($N = 6$) positioned along a circle. The permutations are ordered by a lexicographic ordering, where $\pi_1 \leqslant \pi_2$ means that there exists an index value $i_0 \in 1 \ldots N$ such that for every $i < i_0$, the node $n_i$ is on the same position in both permutations and for the element $n_{i_0}$, the position in $\pi_1$ is less than in $\pi_2$. The lines between two permutation nodes represent the neighborhood relationship. Two permutations are neighbored if they can be converted into each other by a single inversion transformation. The red ray lines from the centre represent the goal function (the route length) of the corresponding permutation. The largest route length has a zero ray length, while the longest ray segment denotes the minimum route length.

If we allow discovering the whole neighbourhood in the heuristic optimization process, then we get 16 local optima denoted by green points in the Figure 3. The number of neighbouring elements is $\binom{N}{2}$, thus the processing of the elements in the neighborhood can be very time consuming for large problems.

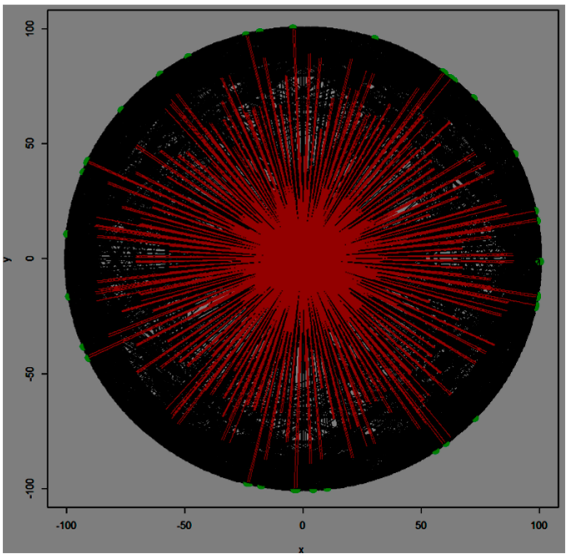



**Figure 3.** TSP Optimization landscape (N = 6).

In the proposed method, the tour planning for a cluster is performed with the following algorithm. Having an initial tour $t_V^0$ and the $V'$ vertex subset, the $\pi^{t_V^0}$ route can be segmented into inner-$V'$ and outer-$V'$ sections. The start and end vertexes of the outer-$V'$ sections are taken as border vertexes. For the local route optimization, the set of $V'$ vertexes are extended with the border vertexes. In the tour optimization of the local cluster, the link between two related border vertexes denotes an external section which cannot contain inner vertexes. Thus the related border vertexes must be adjacent in the winner optimal local Hamiltonian cycle. This constraint is ensured with the following adjustment of the distance matrix. The distance value between two related border vertexes is set to 0 (or very near to zero), thus this edge will be included into the optimal tour with high probability. The system always verifies this constraint explicitly and the proposed tour is rejected if the route is invalid.

After generating the local optimal tour, it will be extended with the corresponding outer sections. For every related border vertex pair, there exists a sub-tour from the global tour and the edge of the border vertex pair will be substituted with the corresponding sub-tour. In an extreme case, the sub-tour may be empty and the two border vertexes denote the same node, that is, the external section contains only one vertex.

The corresponding algorithm (see Algorithms 1 and 2) of the proposed cluster level refinement proceeds as follows:

**Algorithm 1**: Ic_optim (V, T)

```
// V: set of all vertexes
// T: current optimal tour
// l: length of current tour
// W: set of clusters
// Tc: candidate tour
l : = d(T); // calculate tour length
W := gen-clusters(V);    // generate clusters
for each VL in W do    // loop on clusters
    // refinement of the local route
    Tc := Intra-cluster_reordering (V, VL, T);
    lc = d(Tc); // calculate tour length
    if lc < l then
        T := Tc;
        l : = lc;
            end if;
end for;
```

**Algorithm 2**: Intra-cluster_reordering (V, VL, T)

```
// V: set of all vertexes
// VL: set of cluster vertexes
// T: current optimal tour
// B: set of border vertexes
// ET: set of external tour segments of T
// VLB: the extended local vertex set for local optimization
// DL: adjusted distance matrix for the VLB set
// TL: the local optimal tour
(B, ET) : = partition_tour (T, VL); // determine inner and outer sections
VLB := VL union B; // replace outer sections with port symbols
DL := gen_dist(VLB); // generate distance matrix
TL := Optim (VLB, DL); // generate optimal local route
Tnew : = merge (T, TL, B); // update global route
Return (Tnew); // the new global optimal tour
```

## 4. Cost Analysis of the Cluster Level Refinement Method

In the cost analysis of the tour construction algorithm using cluster level refinement (*intra-cluster_reordering*), the cost factors depend on the following parameters:

$N$: number of vertices in $V$;
$f$: cost function of the local optimization algorithm;
$N'$: number of vertices in local cluster $V'$;

In the *partition_tour* function, the tour $T$ is segmented into $V'$ inner and $V'$ outer sections. Having a list representation of the tour and using a status field in the vertex descriptor, the cost of this step can be approximated with

$$O(N).$$

Regarding the generation of the distance matrix, there are two main approaches to be applied. In the first version, there is a global distance matrix generated in the preparation phase of the optimization algorithm. This step requires

$$O\left(N^2\right)$$

cost, while the other approach does not use a global distance matrix but it calculates the local distance matrix in each iteration step. The cost of this step is equal to

$$O\left(N'^2\right).$$

The generation of the local optimum route is performed with a cost value

$$O(f(N')).$$

In the last phase, the local optimum tour is merged with the current global optimum tour. The cost of this operation is equal to

$$O(N).$$

In the case of application of global distance matrix, the total cost can be given with

$$O\left(N^2 + m \cdot \left(N + f\left(N'\right)\right)\right),$$

where $m$ denotes the total number of iterations. Considering a partitioning with $M$ clusters, the approximation for the cluster size is

$$N' \;=\; \frac{N}{M}.$$

The value of $M$ related to the optimum cost can be calculated with the following formula for the case of global distance matrix:

$$c_1 N + c_2(f(\frac{N}{M}) - \frac{N}{M} f'(\frac{N}{M})) \;=\; 0.$$

When using local distance matrix, the corresponding equation is

$$c_1 N - c_2 \frac{N^2}{M^2} + c_3(f(\frac{N}{M}) - \frac{N}{M} f'(\frac{N}{M})) \;=\; 0.$$

Using the approximation

$$f(x) \;=\; x^{\alpha}.$$

where $\alpha > 1$. The optimal cluster number for the global distance matrix approach is given by

$$M_o \;=\; N \cdot \sqrt[\alpha]{\frac{c(\alpha - 1)}{N}}.$$

The optimal relative size of the clusters depends on both the $\alpha$ parameter and the N value. Taking the simplification assumption that all the cost coefficients are equal to 1 ($c_1 = 1, c_1 = 1, \ldots$ ), we can calculate the optimal values. The dependency between $\alpha$ and the optimal cluster size is shown in Figure 4, where the curve denoted with hollow circle is for $N = 100$, filled circle for $N = 1000$ and rectangle symbol is for $N = 10{,}000$.

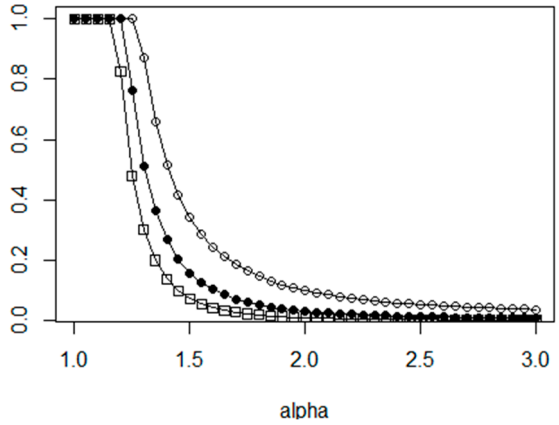



**Figure 4.** Optimal relative cluster size in dependency of alpha using global distance matrix.

In the case of local distance matrices, the optimal $M_o$ value is calculated as the solution of the following equation:

$$c_1 N - c_2 \frac{N^2}{M^2} + c_3(1-\alpha)\left(\frac{N}{M}\right)^{\alpha} = 0.$$

In Figure 3, the optimal relative size of the clusters is shown for two $N$ values, the line with hollow circle notation belongs to $N = 100$, the line with filled circles relates to $N = 1000$. As the figures show, in the case of local distance matrix management, the optimal cluster sizes are smaller than the optimal sizes of global distance matrix management.

Considering the time cost of a global optimization and of the optimization for the partitioning with optimal cluster size, we get the ratio function presented in Figure 5. The function shows the ratio value $r = \frac{cost_{partition}}{cost_{global}}$ for different $\alpha$ values. Based on the test calculations, we can say that the partitioning method provides benefits especially for larger problems using base optimization method with higher execution costs.

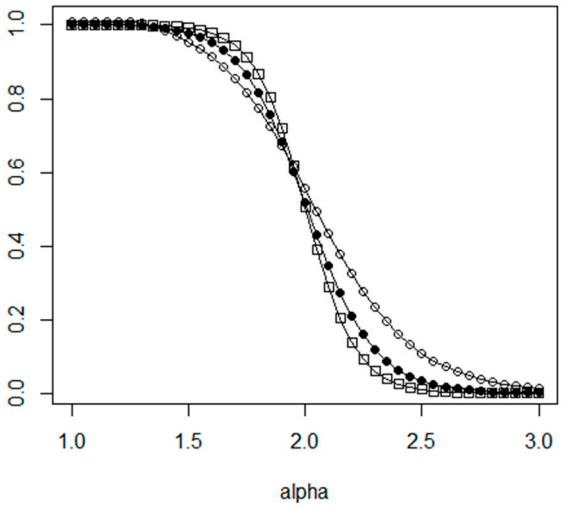



**Figure 5.** Cost ratio in dependency of $\alpha$.

Considering the construction of clusters, there are many different clustering approaches in the literature. In our algorithm, the quality threshold clustering method was implemented as it has many benefits from the viewpoint of the sTSP problem. Based on the obtained experimental results, we can say that the adjacent elements in the route are usually the nearest neighbours in the object space, too. We have investigated optimal tours and tested whether adjacent elements of the route are nearest neighbours in the node space or not. In Figure 6, the histogram of the adjacent element's position in

the corresponding neighbourhood is shown. The axis X denotes the position in the neighbourhood and axis Y denotes the corresponding frequency in the set of adjacent elements. In the test $N$ is set to 4000. According to our test results, about 75% of the adjacent elements are the nearest neighbour elements, too. Thus the clusters must contain elements which are in the near neighbourhood of each other. The cliques generated by the QTC algorithm can provide this property as it contains such elements that for every pair, the distance is always less than a given threshold:

$$C_{d_0} = \{x \mid \forall x, y \in C_{d_0} : d(x, y) \leq d_0\}$$

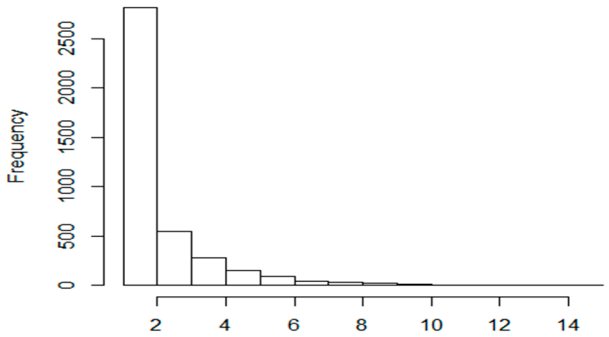



**Figure 6.** The relative distance of the adjacent elements in the optimal tour.

## 5. Performance Evaluation Tests for Local Refinements

In the following, the efficiency of the proposed intra-cluster level route improvement approach is investigated. The main question is to what extend can the intra-cluster level reordering improve the shortest tour found so far. For the performance analysis, a series of tests were executed.

Our tests focused on the operations where the proposed cluster level refinement method may provide an improvement against the competitor methods. There are three main areas where IntraClusTSP can be applied to improve the optimization performance:

(a) General refinement of initial tour using arbitrary internal/local TSP optimization method;
(b) Application of refinement for a specific local area;
(c) Incremental route generation.

In the case of incremental route generation, an existing node graph with optimal route is extended with a new node. The task is to generate a new optimal Hamiltonian cycle involving the new node. One solution is to consider the new graph as an independent new task and we use some existing batch TSP optimization method. This solution requires a lot of redundant calculations as most part of the tour remains unchanged. The incremental approach will perform only the required modifications on the initial route, thus its time cost is significantly lower. Among the known TSP optimization method, the NN algorithm is based on this incremental approach but it tests only a limited set of neighbours to find best position of the new node. From this point of view, the IntraClusTSP method can be considered as an improved version of the standard NN optimization method.

Based on the detailed analysis and performance comparison in Reference [21,28,51], the following methods were implemented for route generation in the performance tests: (a) Nearest Neighbour direct construction algorithm (NN); (b) GA-based refinement algorithm (GA); (c) Hierarchical GA-based refinement algorithm (HGA); (d) Nearest insertion algorithm with refinement (N2); and (e) Chained Lin-Kernighan refinement algorithm (LK).

### *5.1. Evaluation Methodology*

We have selected two different node distributions to generate the input data set for the investigated eTSP problem. The first version is the usual uniform data set distribution in the two dimensional

Euclidean space. Most of the benchmark data sets in the well-known TSPLIB library (https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95) have similar distribution characteristics. In this case, the point locations are distributed within a two-dimensional square uniformly. The only input parameter of the input set is the number of points (N). The second distribution model uses two level distribution, that is, there is cluster level containing the cluster centres and there is node level distribution for the point positions within the clusters. The elements of both levels are generated with uniform random distribution. Here, the generation algorithm involves the following parameters: number of the clusters ($M$), number of the points ($N$), radius of the clusters ($r_0$).

In the tests, one execution step corresponds to a series of local refinement runs on all elements of a partitioning. For example, if the domain is partitioned into $M$ clusters, the execution step contains $M$ runs on all of the clusters. In the experiments, we have tested clustering with both non-overlapping and overlapping clusters. The cluster shapes considered were circular with a given centroid and radius. In the tests, a clustering is given with the following parameters. $N$: number of objects, positions; $M$: number of clusters; $r_0$: relative radius of the cluster; $L$: number of iterations.

Regarding the cluster construction for the refinement runs, we have used a QTC algorithm to select a compact set of neighbouring positions. The mean value for the radius of the refinement clusters was set to 15% of the diameter of the base region.

All performance tests were performed in R environment. The Concorde TSP Solver package (http://www.math.uwaterloo.ca/tsp/concorde.html) was used to execute the Chained Lin-Kernighan algorithm. Using this optimization method, we The Nearest insertion algorithm with two_opt refinement was included from the TSP package. For execution of the GA optimization, the implementation of the GA package was invoked into our test program. In the tests, we have selected the method "linkern" for Chained Lin-Kernighan. Based on our experiments which are summarized in Table 1, this method provides the best optimization quality. In the table, TL denotes the tour length value, while T is the symbol for the execution time. Three Lin-Kerninghan variants were compared, beside linkern-variant, also the nearest insertion and the arbitrary insertion variants were tested. The Nearest Neighbour and the Hierarchical GA-based refinement algorithm were implemented directly.

**Table 1.** Comparison of the different Lin-Kernighan variants.

| N | TL Linkern | TL Nearest | TL Arbitrary | T Linkern | T Nearest | T Arbitrary |
|---|---|---|---|---|---|---|
| 600 | 18 | 23 | 20 | 1 | 2 | 1 |
| 1200 | 25 | 32 | 29 | 1 | 6 | 1 |
| 1800 | 31 | 39 | 35 | 2 | 36 | 1 |
| 2400 | 35 | 44 | 40 | 2 | 90 | 1 |
| 3000 | 39 | 50 | 45 | 3 | 150 | 2 |

### *5.2. Nearest Neighbour Direct Construction Algorithm*

The goal of the first experiments was to evaluate the efficiency of IntraClusTSP refinement algorithm for routes generated by the Nearest Neighbour method. The Nearest Neighbour direct construction method has only one main algorithmic parameter, the distance matrix of the graph. In our case, we have constructed the distance matrix from a point distribution using the Euclidean distance. In the test, we have generated the initial route with the NN method, then we have applied a sequence of IntraClusTSP refinement steps. The results for the data set parameters $M = 324$, $r_0 = 0.15$ on a sample with $N = 6000$ are shown in Figure 7a,b. According to the test results, the proposed cluster route refinement algorithm improved the route length by 8%. As it is shown in the figure, the first refinement cycle provides the largest improvement (about 5%). Based on the literature, the tour length generated by the NN algorithm is about 25% above the theoretical optimum value, the result of the proposed refinement is about 14% above the theoretical optimum. Considering the comparisons performed in Reference [29], this result is a significant improvement of the original method. The optimal route length as a function of the iteration count is shown in Figure 4. Considering the number of generated

clusters, *M*, we can see that the IntraClusTSP method provides a better improvement for partitioning with higher number of clusters.

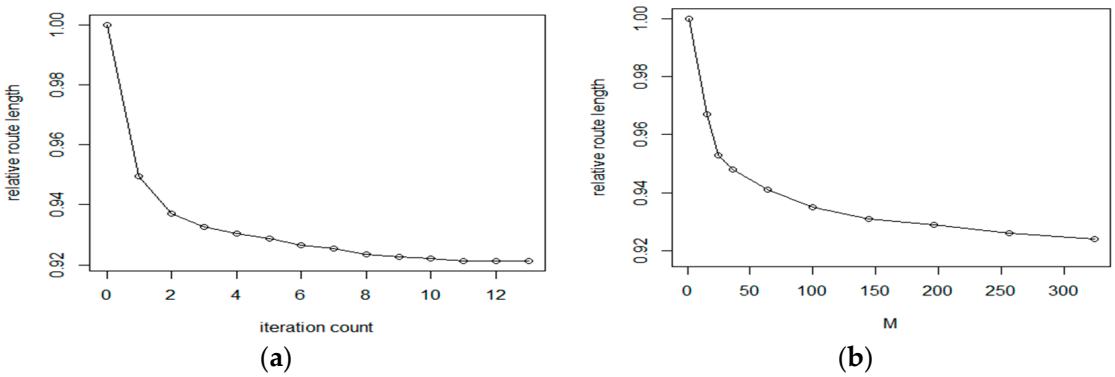



**Figure 7.** (**a**,**b**) Efficiency of improvement iterations for the NN algorithm.

In Figure 8, two time cost functions are presented. The first function corresponds to the base NN method (denoted by hollow circles) while the second shows the execution cost of IntraClusTSP (filled circle). The second function corresponds to the case when M is equal to the calculated optimum value. As it is expected, the total cost of optimizing the whole node set is higher than the cost to perform a set of local optimizations for the covering cluster set.

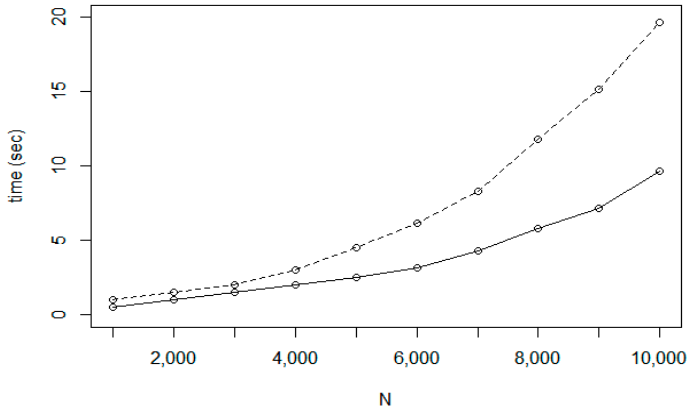



**Figure 8.** Time cost of the base NN and of the IntraClusTSP refinement method (solid line).

### *5.3. GA and Hierarchical GA Refinement Algorithms*

In the GA based refinement algorithm, an individual route corresponds to a permutation of the positions. The fitness of an individual is given with the corresponding tour length. The GA algorithm uses the crossover, mutation and selection operators on the population of selected individuals. Based on our experiences, if the number of population iterations is limited, the GA can provide only weaker results, especially for GA with random initialization. In order to improve the search space, the route of the NN direct optimization algorithm is taken as an element of the initial population. In the case of hierarchical GA, the algorithm first performs a partitioning of the original position set into disjoint clusters. In the next step, every cluster is considered as position represented by the centre point and the optimal route on the cluster level is calculated. Then, for every cluster, a local tour optimization is performed and finally the local optimal routes are merged into a global optimal route.

Based on the experiences, as it is shown in Figure 9, the random GA is significantly dominated by the NN-initialized GA method. In the figure, the values related to the random GA are given with hollow circles, while NN-initialized GA is denoted by filled circles. The GA algorithm is based on the following main parameters: population size, probability of crossover, probability of mutation,

maximum number of iterations to run before the GA search is halted and number of consecutive generations without any improvement in the best fitness value before the GA is stopped. Based on our preparation test, the population size has the largest effect on optimization efficiency. For a wide range of N in our input, the optimal population size is near 40, we have used this parameter in our comparison tests. Regarding the other parameters, the following settings was used: probability of mutation: 0.2, probability of crossover: 0.7; maximal number of iterations: 500 and number of runs: 200.

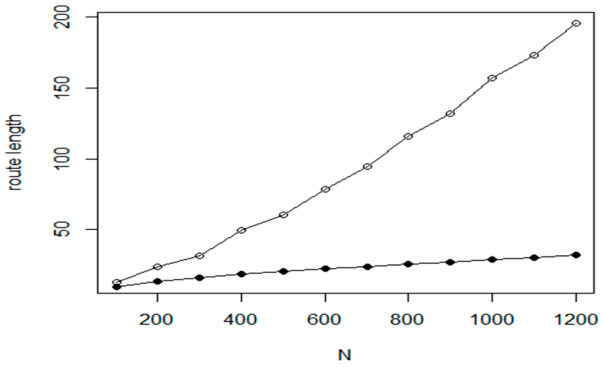



**Figure 9.** Route length of the random GA and the NN-GA methods (filled circle).

The results of the NN_GA can be improved by some percent using the hierarchical GA approach. The corresponding results are given in Figure 10, where the filled circles denote the HGA method. The proposed refinement method can reduce the tour length by 5–10%, similar to the base NN approach (see Figure 11).

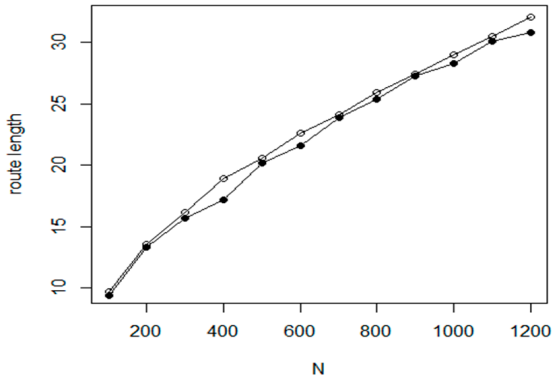



**Figure 10.** Route length of the NN-GA and the HGA (filled circle) methods.

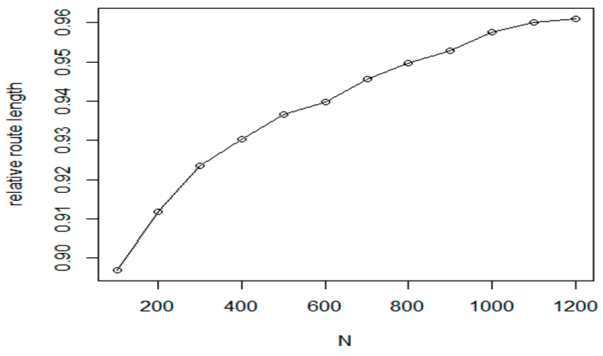



**Figure 11.** Efficiency of improvement iterations for the HGA algorithm.

### *5.4. Nearest Insertion Algorithm with Two_Opt Refinement*

The Nearest insertion algorithm with two_opt refinement is the default solver in the TSP package of R and it can provide a very good approximation of the optimal route. The Nearest insertion (NI) algorithm chooses city in each step as the city which is nearest to a city on the tour. The NI and two-opt algorithms does not require any special parameter to be set in our tests. Figure 12 shows the measured efficiency comparison of the Nearest Neighbour method and the Nearest insertion with two_opt refinement algorithm. In the figure, the following notations are used.

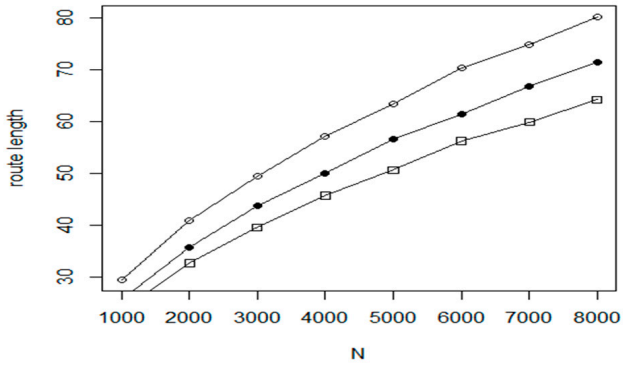



**Figure 12.** Efficiency of the Nearest insertion algorithm with two_opt refinement.

The hollow circle denotes the Nearest Neighbour method, filled circle is for Nearest insertion algorithm with two_opt refinement and hollow rectangle is the symbol for the theoretical optimum. Although the efficiency is very good, we experienced a weakness of this algorithm too, the high execution time. As it is shown in Figure 13, the execution time for the problem with $N$ = 8000, the required time is more than 200 times larger with $t_{NN}$ = 10 s and $t_{NI+2opt}$ = 2300 s.

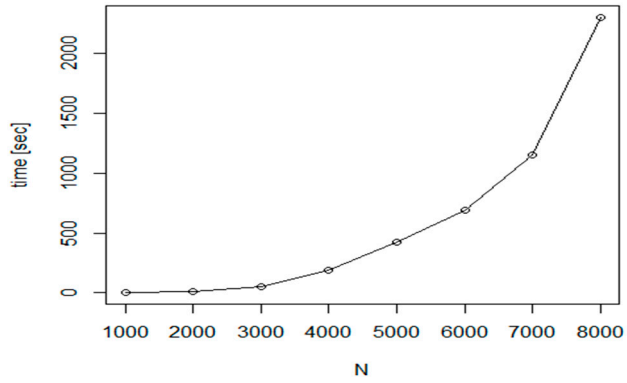



**Figure 13.** Execution cost of the Nearest insertion algorithm with two_opt refinement.

Based on this experiment, the method is not suitable for larger problems. In order to reduce the execution costs, a hybrid approach was introduced in our tests. The proposed method first uses the fast NN algorithm to create an initial route. Then, in the refinement phase, the Nearest insertion algorithm with two_opt refinement is used for tour optimization at cluster level. Taking an iteration of cluster level execution steps, we have created a fast and efficient algorithm.

The experiments prove that the proposed method can improve the efficiency of base method by 2–3% (see Figure 14) while the required execution time is only a small fraction of the base execution time (Figure 15). In the experiments, the following parameter values were used: $M$ is between 25 and 64; $L$: between 6 and 10 and $r_0$ is equal to 0.15.

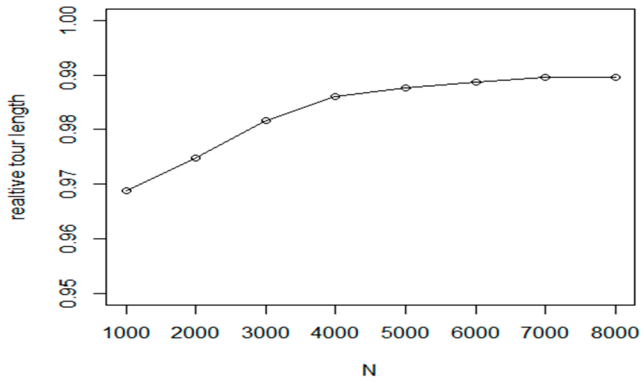



**Figure 14.** Relative efficiency of the proposed NN-initialized NI-2opt cluster level refinement method.

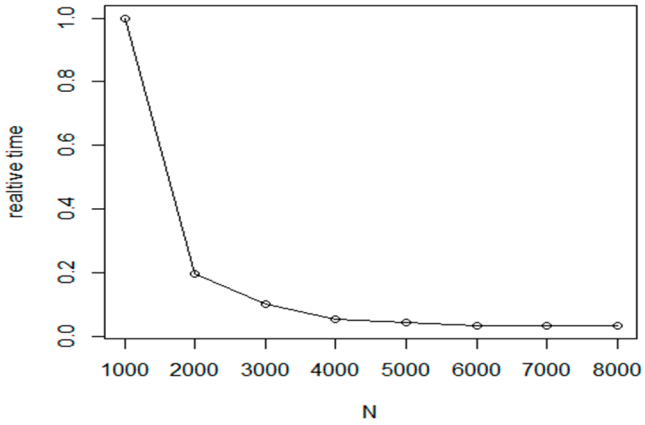



**Figure 15.** Relative time cost of the NN-initialized NI-2opt cluster level refinement method.

### *5.5. Chained Lin-Kernighan Refinement Algorithm*

According to the analyses [19,28], the Chained Lin-Kernighan refinement algorithm provides the best solution for the TSP optimization problems. Its tour length efficiency is very near to the theoretical optimum (the difference is only 1–2%). This superior efficiency is confirmed by our test experiments too. In our tests, the LINKERN package was used to run the Chained Lin-Kernighan refinement algorithm. The Chained Lin-Kernighan refinement method is an algorithm with many parameters. In our tests, we have used the standard settings [52] with the following values: Kick value: random walk; number of kicks: number of nodes; method to generate staring cycle: QBoruvka.

The cluster level refinement method can provide here only a tiny improvement. Based on our test experiments, the average improvement ratio is 0.2%, the best result is a one percent improvement for an input distribution with $N$ = 60,000. Considering the fact that the result of the Chained Lin-Kernighan refinement method is very near (1–2%) to the theoretical optimum, the case with 1% improvement is an important achievement.

### *5.6. Convergence of the IntraClusTSP Refinement Method*

In the tests to analyse the convergence of the proposed IntraClusTSP method, we have selected the GA method with random initialization to generate the initial route. Regarding the convergence efficiency of IntraClusTSP, the main determining factor is the applied algorithm for the cluster-level optimization. The Chained Lin-Kernighan method applied in our proposal, provides a very fast convergence. In Reference [53], the investigation of the route length—iteration step function has shown that in the first 20% of the total running time, the optimization process provides 80% improvement and the rest 80% of the time yields 20% improvement. In our cluster-level refinement approach, the convergence rate is influenced by the following factors:

- size of the cluster (large clusters can provide larger improvements but they require larger execution time)
- quality of the cluster-level route (the quality shows how far is the length of the current route from the optimal length)

As the average quality of the graph increases during the refinement process, the convergence rate will decrease as it is shown also in the referenced paper.

In our convergence tests, we have used random cluster selection. In every iteration step, only one cluster was processed. In the tests, the following parameter settings were used:

$N$ (number of nodes): 400, 800, 1200
$r$ (relative radius of the clusters): 0.1, 0.2, 0.3
$m$ (number of iteration steps): 25

Our test results can be summarized in the following points:

- Similar to the convergence of the base LK method, the convergence ratio is significantly larger in the first few iteration steps. In Figure 16, a sample convergence example is presented as a route length—iteration count function for the input parameters ($N$ = 400, $r$ = 0.2)
- The convergence ratio is larger for larger cluster sizes. Figure 17 shows the comparison run for three different cluster size values. The bottom (solid) line belongs to $r$ = 0.3 (large clusters); the middle (dashed) line relates to $r$ = 0.2 and the top (dotted) line belongs to $r$ = 0.1. The reason of the fact that the curves may have local plateau is that the clusters are selected here randomly, thus there is a chance to select areas already processed before. In this case, only little improvement can be achieved.
- The convergence as relative length reduction is not very sensitive to the graph size but as the experiments show, in larger graphs we can achieve larger length reduction and a better convergence. In Figure 18, the results for three different sizes are summarized. The bottom (solid) line belongs to $N$ = 1200 (large graph); the middle (dashed) line relates to $N$ = 800 and the top (dotted) line belongs to $N$ = 400.

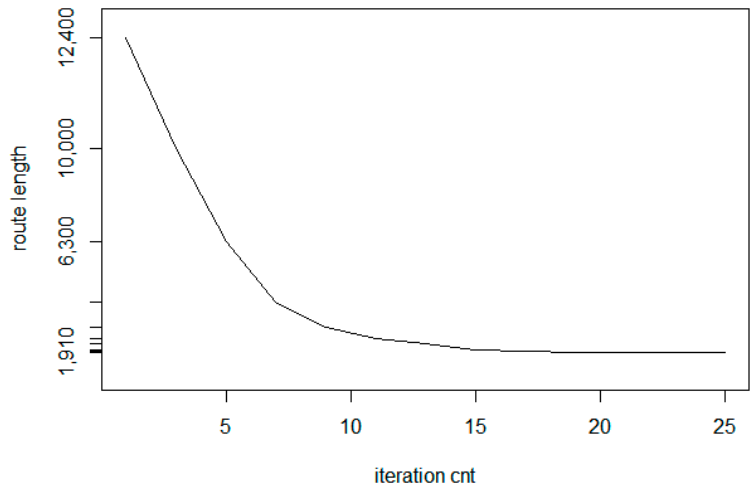



**Figure 16.** Convergence function of the IntraClusTSP algorithm.

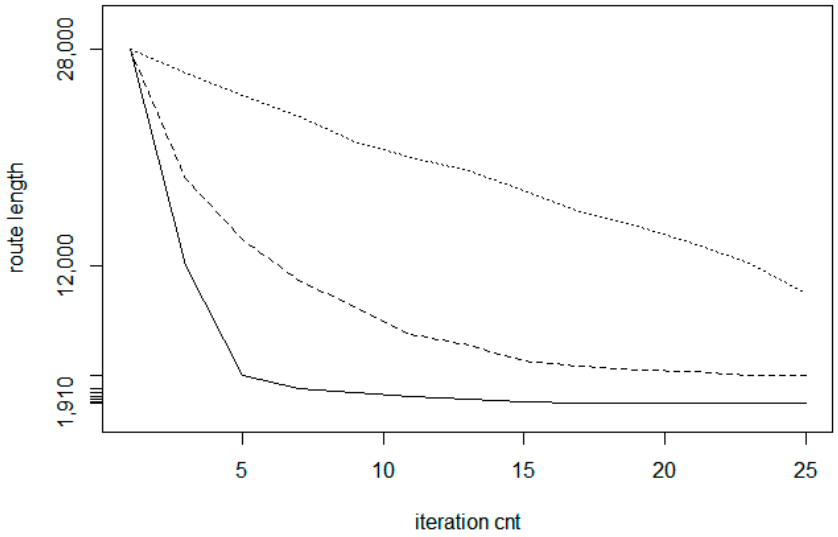



**Figure 17.** Comparison of the convergence for different cluster radius values (solid line: r = 0.3, dashed line: r = 0.2; dotted line: r = 0.1).

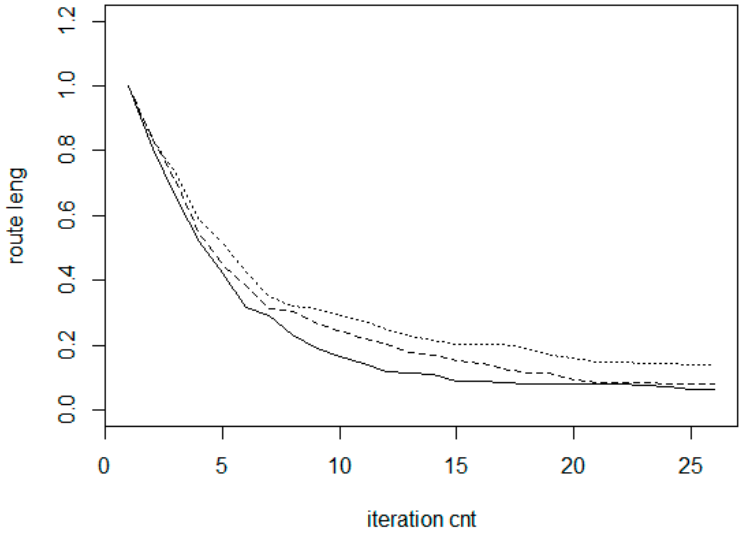



**Figure 18.** Comparison of the convergence for different graph size values (solid line: $N$ = 1200, dashed line: $N$ = 800; dotted line: $N$ = 400).

## 6. Performance Evaluation Tests for Incremental TSP

In the following, we compare IntraClusTSP with Chained Lin-Kernighan method and with the Random insertion methods from the viewpoint of both time cost and route length optimization efficiency. We have involved the following algorithms into the tests:

- Random insertion with random position (RR);
- Random insertion with smallest single distance (RS);
- Random insertion with smallest route length increase (RI);
- Random insertion with smallest single distance with IntraClusTSP refinement (RCI);
- Chained Lin-Kernighan method on the whole graph (LK).

In the case of RR, the position of the new node in the route is selected randomly.

First, we investigate the insertion of a single new node into the current optimal route. Considering the route length optimization, we have experienced that in this simple case, all methods provided about the same result. Thus, there were no significant differences between the investigated methods,

only the RR method had a worst time optimization value. In the tests, the method RR provided a very low efficiency as it is shown in Table 2 and Figure 19. The input node distribution was generated with uniform distribution within a rectangle area. The size of the graph was running from 1000 to 10,000. The proposed IntraClusTSP algorithm provided a better result than known insertion methods. In Figure 19 we present the cost difference between the new and old tour versions for the insertion algorithms and for the Chained Lin-Kernighan method. The presented values are the average values related to samples of size 10.

**Table 2.** Average tour length increase.

| Method | Increase |
|---|---|
| RR | 1253 |
| RS | 17 |
| RI | 11 |
| RCI | 9.3 |
| LK | −1.2 |

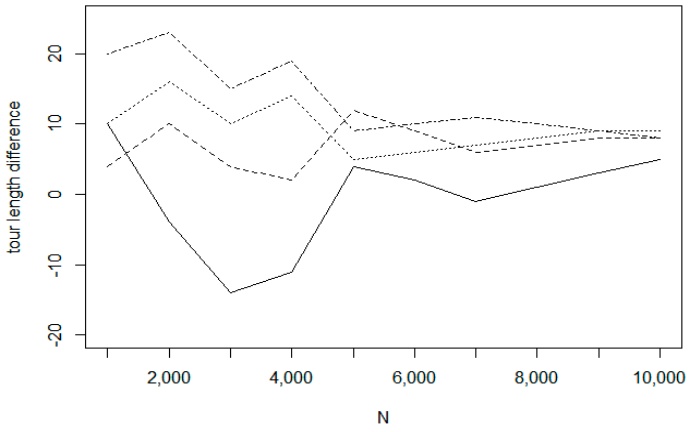



**Figure 19.** Tour length increase functions. (LK: solid; RCI: dashed, RI: dotted, RS: dotdash).

In the next experiments, we investigated the efficiency for insertion sequences, when 200 new elements are inserted into the graph. For the case of single insertion, is proven that our algorithm provides a better result in general but the difference tends to decrease as the size of the initial tour increases. In the case of insertion sequences we may get a different result, as standard insertion methods (RS, RI) can modify only one edge in the original tour, while RCT can perform a larger scale modifications during the same amount of time.

In the tests on insertion sequences, we have selected three kinds of training set. The first is the uniform random distribution (UDT) while the two others (TSPLIB_fin10639 and TSPLIB_xql662) belong to the TSPLIB data repository. For the TSPLIB_fin10369 data set (http://www.math.uwaterloo.ca/tsp/world/countries.html) we performed two experiments with different initial tour sizes. The test results are shown in Figures 20–23. In Figure 20 which relates to the uniform random distribution, the bottom dot-dashed line belongs to the Chained Lin-Kernighan method which is a batch method. For this method, the line is a linear approximation, only the values at the start and end positions are measured. For the incremental methods, all values are measurement results. The second solid line denotes the proposed IntraClusTSP method. The next line relates to the RI (Random insertion with smallest route length increase) algorithm and the weakest result is given by the RS (Random insertion with smallest single distance) method. Similar efficiency order can be observed by the other experiments too. Figure 21 relates to dataset fin10639 on the size interval 5000–5200. Figure 22 shows the tour length values for the size interval 10,000–10,200 of the same data set. In Figure 23, the measured tour length values are presented for the TSPLIB xql662 dataset (http://www.math.uwaterloo.ca/tsp/vlsi/).

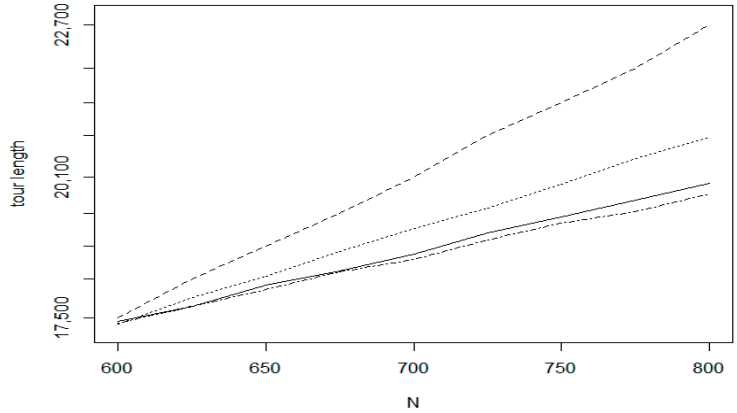



**Figure 20.** Tour length functions (UDT).

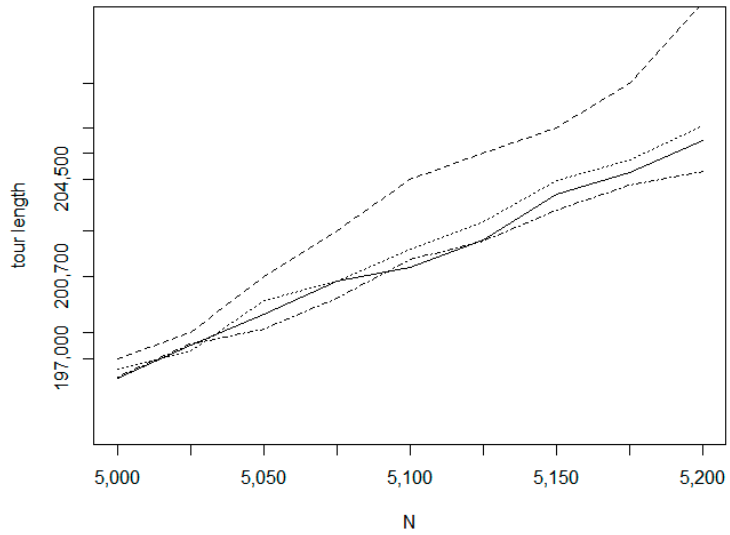



**Figure 21.** Tour length functions (TSPLIB_fin10639).

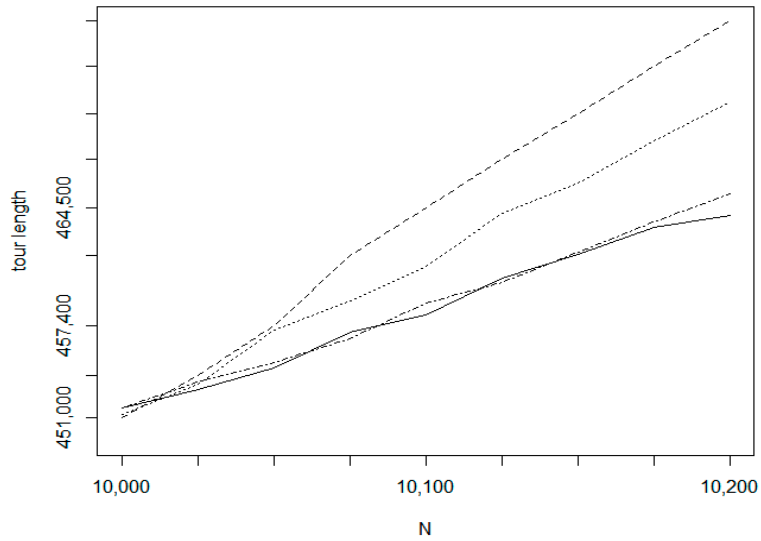



**Figure 22.** Tour length functions (TSPLIB_fin10639).

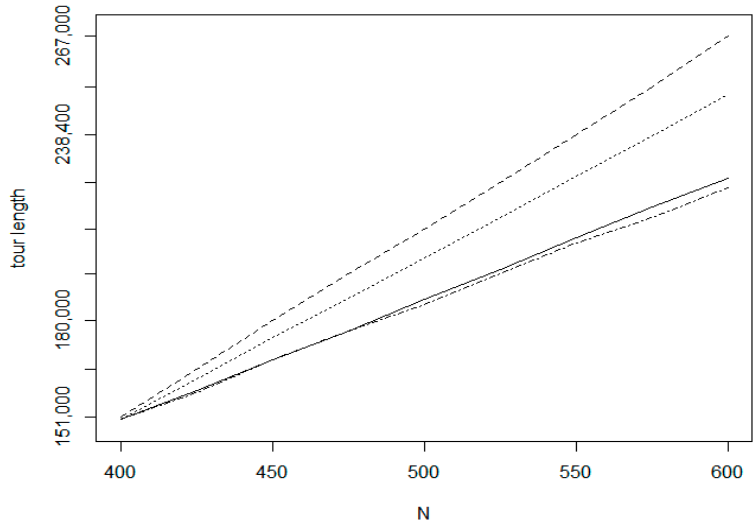



**Figure 23.** Tour length functions (TSPLIB xql662).

In Figure 24, the time costs are compared. The largest cost belongs to the Lin-Kernighan method, the cost function of NN shows a very similar characteristics. The incremental route update using the IntraClusTSP algorithm requires only 0.5 s for the graph with $N$ = 10,000. In case of Random insertion with smallest route length increase algorithm on the fin10369 dataset, required 10 s, which is a significant difference.

The test results proved that the proposed IntraClusTSP significantly dominates the random insertion algorithms on the field of incremental TSP construction, regarding both optimization efficiency and execution costs.

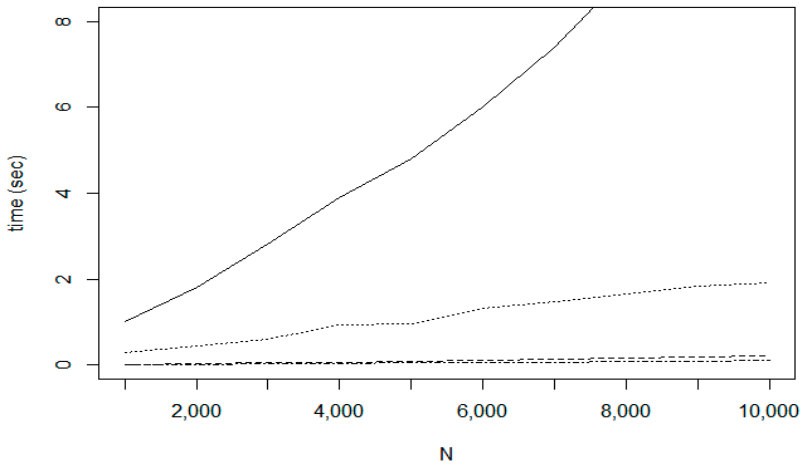



**Figure 24.** Time cost functions (LK: solid; RCI: dashed, RI: dotted, RS: dotdash).

## 7. Application in Data Mining

The proposed IntraClusTSP algorithm can be used also in the context of data analysis to support clustering A cluster group consists of elements similar to each other. A key factor in clustering is the quality of the clustering (how similar are the objects within a cluster to each other) [54] and the number of the clusters. Usually, the users should set this number either directly or indirectly in advance as an input parameter of the clustering process. For example, in the case of k-means clustering, the number of required clusters is fixed in advance. The determination of the appropriate number of clusters, is considered as a fundamental and largely unsolved problem [50]. In the literature, we can

find numerus approaches like Silhouette statistics [55], gap-statistics [56] or Gaussian-model based approach [57].

In this section, we present an algorithm based on TSP optimization to support the determination of the optimal number of clusters. The proposed method is based on the following considerations. As it was shown in Section 3, the shortest path usually connects the elements located in the neighbourhood. Thus, the distances between adjacent elements belonging to the same cluster, should be small. On the other hand, if the adjacent element belongs to different cluster, the distance should have a large value. Thus, a relatively large distance in the optimal route should mean a gap, a connection from a cluster to another cluster. Based on these considerations, we evaluate the distance function between two adjacent elements of the optimal route to determine the clustering structure.

The proposed method is based on the analysis of the corresponding edge length histogram. Having this histogram, we consider it as a mixture of normal distributions and we can perform a Gaussian Mixture Density Decomposition (GMDD) [58]. In GMDD, having the current density function, we first determine the location of the maximum density and we fit a Gaussian distribution with maximum overlap. This Gaussian is added to the components' list and this component is removed from the current input distribution. If this reduced density is a not a zero function then it will be analysed in the same way. Figure 25 shows the result of a sample GMDD process.

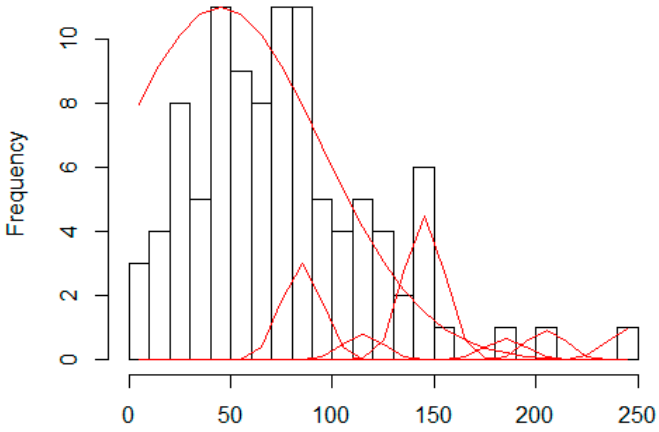



**Figure 25.** Sample GMDD decomposition.

Having the GMDD spectrum, we have to determine those components which belong to different clustering levels. In our model, we consider an edge as cluster level connection if its length is significantly larger than the average length of the intra-cluster connections:

$$l_c > \propto \bar{l}_i$$

The parameter $\propto$ is considered as an input parameter of the algorithm. Based on our experiments with human observers, we take value 2 as default value of alpha. Our tests show that the point distribution in Figure 26a ($l_c = \bar{l}_i$) is usually considered as single cluster while the points in Figure 26b ($l_c = 2\,\bar{l}_i$) are considered to belong to two clusters.

(**a**) (**b**)

**Figure 26.** (**a**) single cluster; (**b**) two clusters.

Thus, if the length difference between two elements in the corresponding edge length histogram is significantly larger than the average length of the previous level, then the larger element will belong to a new clustering level. The algorithm to calculate the cluster counts for different levels can be summarized in the following steps.

**Algorithm 3**: Histogram of edge length containing M bins

```
1: CL := cluster level descriptors;
2:   g : = 0; // cluster level index
3:   for each i in (1 to M) do
4:     b := H[i]; // current bin
5:     b.count := count of edges in the bin
6:     b.length := average edge length in the bin
7:     bp := the previous not empty bin
8:     if b.count > 0 then
9:         L_g := the average edge length in the current cluster level CL[g];
10:        L_a : = b.length – bp.length;
11:        if L_a > alpha * L_l then
12:           g = g + 1;   //create a new cluster level
13:           add b to CL[g];
14:        else
15:           add b to CL[g];
16:        end if
17:        bp : = b;
18:    end if
19:  end for
20:  Calculate the GMDD distributions for the cluster levels
```

In the following, we present the application of the algorithm on an example distribution.

Figure 27 denotes the initial distribution of the graph nodes. Can be detected two clustering levels. At the first level, there are 12 small clusters, while the second level contains three large clusters. The calculated shortest path is denoted by red line. The generated edge length histogram with the GMDD spectrum is presented in Figure 28, where the corresponding edge-length function is given in Figure 29. The first large peak belongs to the base level, here we have 120 elements in the distribution. The second peak denotes the first cluster level with the small clusters and the last component relates to the level of the large clusters. Table 3 summarizes the main parameters of the discovered clusters.

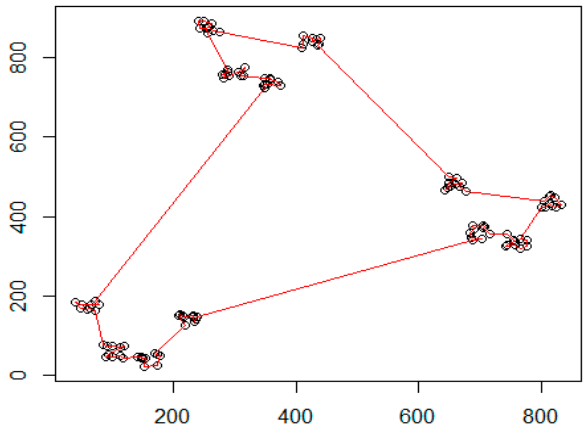



**Figure 27.** Sample initial distribution in a two dimensional space.

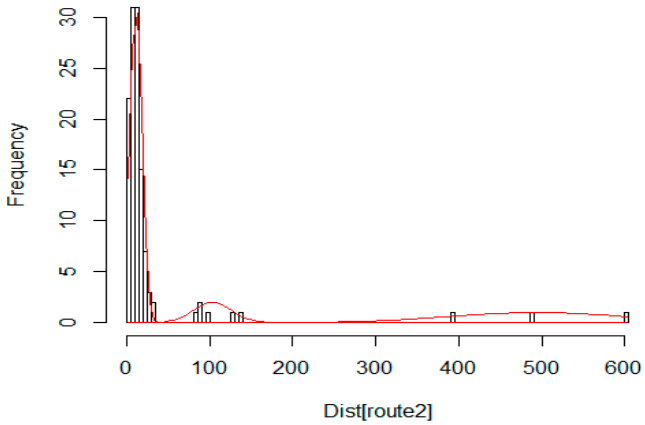



**Figure 28.** GMDD components for the sample distribution.

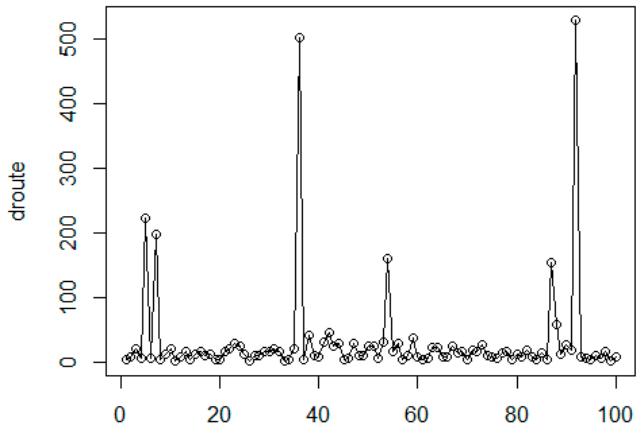



**Figure 29.** The edge length function of the route.

**Table 3.** Cluster level parameters.

| Parameter | Level 0 | Level 1 | Level 2 |
|---|---|---|---|
| Mean length | 10.8 | 81.4 | 497.5 |
| Standard deviation | 5.7 | 33.5 | 33.5 |
| Size | 120 | 12 | 3 |

As the results show the method discovered the multi-level clustering structure in the input distribution.

We have compared our algorithm with two widely used methods for determining cluster count. The first investigated method uses the Silhouette index [59] approach. This index prefers small intra-cluster distances and it is defined in the following way for a partitioning with clusters $(C_1, \ldots, C_k)$:

$$S_k = \frac{1}{N} \sum_{i=1}^{k} S_i$$

$$S_i = \frac{b_i - a_i}{\max(b_i, a_i)}$$

$C'_i$ : the parent cluster of object $i$

$d(i, C')$ : average dissimilarity of object $i$ to all objects in cluster $C'$

$$a_i = d(i, C'_i)$$

$$b_i = min_{C \neq C'_i} \{d(i, C)\}$$

The cluster count is the $k$ value, where the value $S_k$ is maximum optimum:

$$K = argmax_k\{S_k\}$$

The second investigated method is the gap statistic method [55] Estimating the number of clusters in a data set via the gap statistics. The method calculates the goodness of clustering measure by the "gap" statistic.

The $E[\log(W_k)]$ value is calculated via bootstrapping, that is, simulating from a reference distribution. A sample calculated gap statistic value in dependency of cluster count $k$ is shown in Figure 30. There are different approaches to determine the cluster count $K$ from the gap function, like the first maximum or the global maximum.

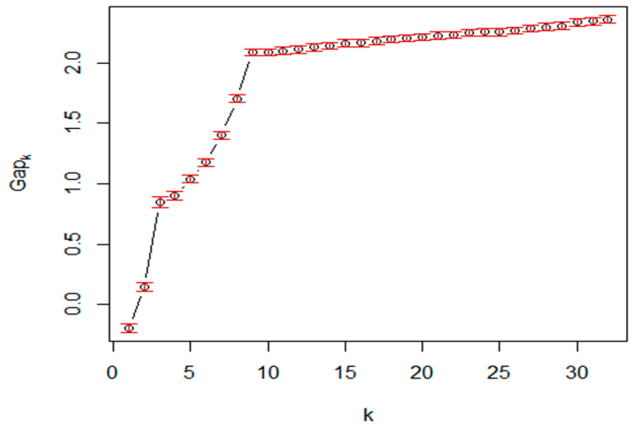



**Figure 30.** The gap statistics function.

In the comparison tests, the factoextra and cluster packages of R were included to perform the silhouette and gap statistic calculations. In the generated data distributions for the evaluation test, the objects are arranged into either a single-level or a two-level clustering structure. For the comparisons, we have generated Table 4 describing the typical test results has the following structure:

1. column (observed): the real clustering structure observed by humans. The column contains one or two numbers. The first number denotes the number of clusters at the first clustering level, while the second value is the cluster counter of the second level.
2. column (gap-A): cluster count proposed by gap analysis using the first maximum optimum criteria;
3. column (gap-B): cluster count proposed by gap analysis using the Tibs2001Semax optimum criteria;
4. column (silhouette): cluster count given by the Silhouette method;
5. column (multi-level GMDD): cluster count calculated by our proposed multi-level GMDD-based method;
6. column (time [silhouette]): the execution time of the Silhouette method in milliseconds;
7. column (time [gap]): the execution time of the gap statistics method in milliseconds;
8. column (time [ml GMDD]): the execution time of our proposed multi-level GMDD method in milliseconds.

**Table 4.** Comparison of the calculated optimal number of clusters.

| Observed | Gap A | Gap B | Silhouette | Multi-Level GMDD | Time [Silhouette] | Time [Gap] | Time [mL GMDD] |
|---|---|---|---|---|---|---|---|
| 12 + 3 | 14 | 3 | 5 | 12+3 | 52 | 2890 | 1 |
| 12 + 3 | 14 | 3 | 4 | 12+3 | 57 | 2991 | 1 |
| 1 | 32 | 3 | 3 | 1 | 52 | 1850 | 1 |
| 1 | 32 | 3 | 3 | 1 | 53 | 1859 | 1 |
| 5 | 20 | 11 | 4 | 5 | 54 | 1920 | 1 |
| 5 | 32 | 7 | 2 | 7 | 54 | 1950 | 1 |
| 5 | 20 | 11 | 4 | 5 | 42 | 2010 | 1 |
| 6 | 32 | 9 | 2 | 6 | 56 | 1980 | 1 |
| 6 + 2 | 32 | 7 | 3 | 7 + 2 | 47 | 1940 | 1 |

The test results in the comparison of the alternative methods can be summarized in the following experiences:

- Only our proposed method can discover the multi-level clustering structure, while the current standard methods are aimed at determining the cluster count of the first clustering level.
- In most cases, the proposed method could discover the hidden clustering structure.
- Our proposed method could provide the best accuracy in all test cases.
- The proposed method requires significantly less execution time.

## 8. Conclusions

The proposed cluster level tour refinement method called IntraClusTSP provides a novel approach to improve the existing heuristic eTSP solution methods. IntraClusTSP takes an initial tour and then performs tour optimization for a selected set of node clusters. In the last phase, the local optimal tours are then merged into a global optimal tour. Based on the performed evaluation tests the proposed method can improve the tour efficiency for every tested base methods (Random insertion with random position; Random insertion with smallest single distance; Random insertion with smallest route length increase; Random insertion with smallest single distance with IntraClusTSP refinement; Chained Lin-Kernighan method on the whole graph). IntraClusTSP performs intermediate level optimizations requiring smaller execution costs than a global level optimization. The proposed IntraClusTSP method can be used to

- refinement of existing routes;
- extension of any existing TSP optimization methods;
- perform a region-level refinement in large graphs;
- implement an efficient incremental route construction method.

For these cases, the proposed method can improve not only the tour length but also the execution time. The generated shortest Hamiltonian path can be used also in data analysis to determine the optimal number of clusters. Our proposed method is based on a novel approach, an adjusted GMDD analysis to determine gaps in the corresponding edge length histogram. Unlike the standard methods, the proposed method can discover also multi-level clustering structure and provides an outstanding performance both in accuracy and execution time.

The proposed cluster level tour refinement method provided very good efficiency and execution time characteristics in our base level experiments. Our plan is to perform further analysis to

- investigate the application for real life TSP problems;
- adapt the proposed algorithm to specific TSP problems like multi-level vehicle routing problem in logistics;
- discover the deeper behaviour of the corresponding permutation space.

**Author Contributions:** L.K. proposed the research direction, participated in the conceptualization, conducted experiments and wrote the paper L.B.I. participated in conceptualization and wrote the paper; D.K.I. participated in conceptualization and wrote the paper.

**Funding:** This research received no external funding.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Zhao, C.; Parhami, B. Symmetric Agency Graphs Facilitate and Improve the Quality of Virtual Network Embedding. *Symmetry* **2018**, *10*, 63. [CrossRef]
2. Jiang, W.; Zhai, Y.; Zhuang, Z.; Martin, P.; Zhao, Z.; Liu, J.B. Vertex Labeling and Routing for Farey-Type Symmetrically-Structured Graphs. *Symmetry* **2018**, *10*, 407. [CrossRef]
3. José, M.V.; Zamudio, G.S. Symmetrical Properties of Graph Representations of Genetic Codes: From Genotype to Phenotype. *Symmetry* **2018**, *10*, 388. [CrossRef]
4. Ball, F.; Geyer-Schulz, A. How Symmetric Are Real-World Graphs? A Large-Scale Study. *Symmetry* **2018**, *10*, 29. [CrossRef]
5. Akiyama, T.; Nishizeki, T.; Saito, N. NP-completeness of the Hamiltonian cycle problem for bipartite graphs. *J. Inf. Process.* **1980**, *3*, 73–76.
6. Laporte, G. The Traveling Salesman Problem: An overview of exact and approximate algorithms. *Eur. J. Oper. Res.* **1992**, *59*, 231–247. [CrossRef]
7. Dantzig, G.B.; Fulkerson, D.R.; Johnson, S.M. Solution of a large-scale traveling-salesman problem. *Oper. Res.* **1954**, *2*, 393–410. [CrossRef]
8. Miller, C.E.; Tucker, A.W.; Zemlin, R.A. Integer programming formulations and traveling salesman problems. *J. Assoc. Comput. Mach.* **1960**, *7*, 326–329. [CrossRef]
9. Baldacci, R.; Hadjiconstantinou, E.; Mingozzi, A. An exact algorithm for the traveling salesman problem with deliveries and collections. *Netw. Int. J.* **2003**, *42*, 26–41. [CrossRef]
10. Held, M.; Karp, R.M. A dynamic programming approach to sequencing problems. *J. Soc. Ind. Appl. Math.* **1962**, *10*, 196–210. [CrossRef]
11. Bomze, I.M.; Budinich, M.; Pardalos, P.M.; Pelillo, M. The maximum clique problem. In *Handbook of Combinatorial Optimization*; Springer: Boston, MA, USA, 1999; pp. 1–74.
12. Carpaneto, G.; Toth, P. Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Manag. Sci.* **1980**, *26*, 736–743. [CrossRef]
13. Hougardy, S.; Wilde, M. On the nearest neighbor rule for the metric traveling salesman problem. *Discret. Appl. Math.* **2015**, *195*, 101–103. [CrossRef]
14. Monnot, J. A note on the traveling salesman reoptimization problem under vertex insertion. *Inf. Process. Lett.* **2015**, *115*, 435–438. [CrossRef]
15. Schuetz, P.; Caflisch, A. Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Phys. Rev.* **2008**, *77*, 046112. [CrossRef] [PubMed]
16. Borůvka, O. "O jistémproblémuminimálním" [About a certain minimal problem]. *Prácemor. Přírodověd. Spol. vBrně III* **1926**, *3*, 37–58. (In Czech and German)
17. Xiang, Z.; Chen, Z.; Gao, X.; Wang, X.; Di, F.; Li, L.; Zhang, Y. Solving large-scale TSP using a fast wedging insertion partitioning approach. *Math. Probl. Eng.* **2015**, *2015*, 854218. [CrossRef]
18. Mosheiov, G. The travelling salesman problem with pick-up and delivery. *Eur. J. Oper. Res.* **1994**, *79*, 299–310. [CrossRef]
19. Weise, T.; Chiong, R.; Lassig, J.; Tang, K.; Tsutsui, S.; Chen, W.; Yao, X. Benchmarking optimization algorithms: An open source framework for the traveling salesman problem. *IEEE Comput. Intell. Mag.* **2014**, *9*, 40–52. [CrossRef]
20. Genova, K.; Williamson, D.P. An experimental evaluation of the best-of-many Christofides' algorithm for the traveling salesman problem. *Algorithmica* **2017**, *78*, 1109–1130. [CrossRef]
21. Ma, Z.; Liu, L.; Sukhatme, G.S. An adaptive k-opt method for solving traveling salesman problem. In Proceedings of the 2016 IEEE 55th Conference on Decision and Control (CDC), Las Vegas, NV, USA, 12–14 December 2016; Volume 1, pp. 6537–6543.

22. Lin, S.; Kernighan, B.W. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.* **1973**, *21*, 498–516. [CrossRef]
23. Ismkhan, H. Effective three-phase evolutionary algorithm to handle the large-scale colorful traveling salesman problem. *Expert Syst. Appl.* **2017**, *67*, 148–162. [CrossRef]
24. Valdez, F.; Melin, P.; Castillo, O. A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation. *Expert Syst. Appl.* **2014**, *41*, 6459–6466. [CrossRef]
25. Kóczy, L.T.; Földesi, P.T.; Szabó, B. An effective Discrete Bacterial Memetic Evolutionary Algorithm for the Traveling Salesman Problem. *Int. J. Intell. Syst.* **2017**, *32*, 862–876. [CrossRef]
26. Imran, A.; Salhi, S.; Wassan, N.A. A variable neighborhood-based heuristic for the heterogeneous fleet vehicle routing problem. *Eur. J. Oper. Res.* **2009**, *197*, 509–518. [CrossRef]
27. Cook, W.; Seymour, P. Tour merging via branch-decomposition. *INFORMS J. Comput.* **2003**, *15*, 233–248. [CrossRef]
28. Johnson, D.; McGeoch, L. Experimental Analysis of Heuristics for the STSP. In *The Traveling Salesman Problem and Its Variations*; Springer: Boston, MA, USA, 2007; pp. 369–443.
29. Johnson, D.S.; McGeoch, L.A. The traveling salesman problem: A case study in local optimization. *Local Search Comb. Optim.* **1997**, *1*, 215–310.
30. Golden, B.; Bodin, L.; Doyle, T.; Stewart, W., Jr. Approximate traveling salesman algorithms. *Oper. Res.* **1980**, *28*, 694–711. [CrossRef]
31. Azar, Y. Lower bounds for insertion methods for TSP. *Comb. Probab. Comput.* **1994**, *3*, 285–292. [CrossRef]
32. Rosenkrantz, D.J.; Stearns, R.E.; Lewis, P.M. An analysis of several heuristics for the traveling salesman problem. *SIAM J. Comput.* **1977**, *6*, 563–581. [CrossRef]
33. Hurkens, C.A. Nasty TSP instances for farthest insertion. In Proceedings of the 2nd Integer Programming and Combinatorial Optimization Conference, Pittsburgh PA, USA, 25–27 May 1992; Carnegie Mellon University: Pittsburgh, PA, USA, 1992.
34. Cronin, T.M. *Maintaining Incremental Optimality When Building Shortest Euclidean Tours (No. CSWD-92-TRF-0042)*; Army Cecom Signals Warfare: Warrenton, VA, USA, 1990.
35. Mladenović, N. An efficient general variable neighborhood search for large travelling salesman problem with time windows. *Jugoslav J. Oper. Res.* **2016**, *23*, 19–30. [CrossRef]
36. Ariyasingha, I.D.; Fernando, T.G. Performance analysis of the multi-objective ant colony optimization algorithms for the traveling salesman problem. *Swarm Evol. Comput.* **2015**, *23*, 11–26. [CrossRef]
37. Gendreau, M.; Hertz, A.; Laporte, G. New insertion and postoptimization procedures for the traveling salesman problem. *Oper. Res.* **1992**, *40*, 1086–1094. [CrossRef]
38. Chisman, J.A. The clustered traveling salesman problem. *Comput. Oper. Res.* **1975**, *2*, 115–119. [CrossRef]
39. Jongens, K.; Volgenant, T. The symmetric clustered traveling salesman problem. *Eur. J. Oper. Res.* **1985**, *19*, 68–75. [CrossRef]
40. Mulder, S.; Wunsch, D., II. Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks. *Neural Netw.* **2003**, *16*, 827–832. [CrossRef]
41. Applegate, D.; Cook, W. Solving large-scale matching problems. In *Network Flows and Matching: First DIMACS Mathematical Problems in Engineering 7 Implementation Challenge*; American Mathematical Society: Providence, RI, USA, 1993; pp. 557–576.
42. Verhoeven, M.G.A.; Aarts, E.H.L.; Swinkels, P.C.J. A parallel 2-opt algorithm for the traveling salesman problem. *Future Gener. Comput. Syst.* **1995**, *11*, 175–182. [CrossRef]
43. Karp, R.M. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Math. Oper. Res.* **1977**, *2*, 209–224. [CrossRef]
44. Bentley, J.J. Fast algorithms for geometric traveling salesman problems. *ORSA J. Comput.* **1992**, *4*, 387–411. [CrossRef]
45. El-Samak, A.F.; Ashour, W. Optimization of traveling salesman problem using affinity propagation clustering and genetic algorithm. *J. Artif. Intell. Soft Comput. Res.* **2015**, *5*, 239–245. [CrossRef]
46. Phienthrakul, T. Clustering evolutionary computation for solving travelling salesman problems. *Int. J. Adv. Comput. Sci. Inf. Technol.* **2014**, *3*, 243–262.
47. Psychas, I.D.; Delimpasi, E.; Marinaki, Y. Hybrid evolutionary algorithms for the multiobjective traveling salesman problem. *Expert Syst. Appl.* **2015**, *42*, 8956–8970. [CrossRef]

48. Bednarik, L.; Kovács, L. Efficiency analysis of quality threshold clustering algorithms. *Prod. Syst. Inf. Eng.* **2012**, *6*, 1275–1285.
49. Garnier, R.; Taylor, J. *Discrete Matrhematics for New Technolology*; IoP Publisher: Bristol, UK, 2001.
50. Sugar, C.A.; James, G.M. Finding the number of clusters in a dataset: An information-theoretic approach. *J. Am. Stat. Assoc.* **2003**, *98*, 750–763. [CrossRef]
51. Haroun, S.A.; Jamal, B.; Hicham, E.H. A Performance Comparison of GA and ACO Applied to TSP. *Int. J. Comput. Appl.* **2015**, *117*, 28–35. [CrossRef]
52. Richter, D.; Goldengorin, B.; Jäger, G.; Molitor, P. Improving the efficiency of Helsgaun's Lin-Kernighan heuristic for the symmetric TSP. In Proceedings of the Workshop on Combinatorial and Algorithmic Aspects of Networking, Halifax, NS, Canada, 14 August 2007; pp. 99–111.
53. Walshaw, C. *A Multilevel Lin-Kernighan-Helsgaun Algorithm for the Travelling Salesman Problem*; CMS Press: London, UK, 2001.
54. Ma, F.M.; Guo, Y.J.; Yi, P.T. Cluster-reliability-induced OWA operators. *Int. J. Intell. Syst.* **2017**, *32*, 823–836. [CrossRef]
55. Rousseeuw, P.J.; Kaufman, L. Finding groups in data. *Ser. Probab. Math. Stat.* **2003**, *34*, 111–112.
56. Tibshirani, R.; Walther, G.; Hastie, T. Estimating the number of clusters in a data set via the gap statistic. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)* **2001**, *63*, 411–423. [CrossRef]
57. Kass, R.E.; Raftery, A.E. Bayes factors. *J. Am. Stat. Assoc.* **1995**, *90*, 773–795. [CrossRef]
58. Yang, X.; Kong, F.; Xu, W.; Liu, B. Gaussian mixture density modeling and decomposition with weighted likelihood. *J. Men's Health* **2004**, *5*, 4245–4249.
59. Liu, Y.; Li, Z.; Xiong, H.; Gao, X.; Wu, J. Understanding of internal clustering validation measures. In Proceedings of the 2010 IEEE International Conference on Data Mining, Sydney, Australia, 13–17 December 2010; pp. 911–916.


Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 45 (2017) 17–29

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

# MetrIntMeas a novel metric for measuring the intelligence of a swarm of cooperating agents

Action editor: Peter Erdi

Laszlo Barna Iantovics [a,*], Frank Emmert-Streib [b], Sabri Arik [c]

[a] Petru Maior University of Tirgu Mures, Romania
[b] Tampere University of Technology, Finland
[c] Istanbul University, Turkey

Received 4 June 2016; received in revised form 14 March 2017; accepted 24 April 2017
Available online 10 May 2017

**Abstract**

We propose a novel metric called *MetrIntMeas* (*Metric for the Intelligence Measuring*) for an accurate and robust measurement of the difficult problem-solving intelligence of a swarm system. The metric allows the classification if a swarm system belongs to the same class with the systems which have a specific reference intelligence value. For proving the efficiency of the proposed metric we realized a case study on a swarm system specialized in solving a NP-hard problem. As an application of the proposed metric, we present the measurement of the swarm systems' evolution in intelligence. We gave a new definition to the intelligent evolving systems. The evolution of intelligent systems can be verified using the proposed *MetrIntMeas* metric.


*Keywords:* Intelligent system; Collective intelligence; Cognitive system; Measuring machine intelligence; Intelligent evolving system; Computational hard problem

## 1. Introduction

Many real life problems are difficult to be solved by computing systems. Difficulty in a problem solving may consist in aspects, like: the description contains missing or erroneous data; the solving is computationally difficult, it is NP-hard for example (Leeuwen, 1998). The efficient and flexible solving of many computational problems is based on cooperative swarms of agents. Recently, many artificial swarm systems were developed, some of them being used for difficult problem-solving (Mamei, Zambonelli, & Leonardi, 2003; Stradner et al., 2013). Most of them are inspired by the collective intelligence of relatively simple living creatures like the insects. As examples of such designed swarm systems, we mention: ants algorithms (Dorigo, Maniezzo, & Colorni, 1996), bees algorithms (Akay & Karaboga, 2012) and flocking of birds (Feder, 2007). In most artificial swarm systems the individuals from the swarm operate as artificial agents, without cognitive capacity and intelligence. Many researches (Brady, Fisher, Schultz, & Ward, 2014; Johnson et al., 2013; Grosz & Hunsberger, 2006; Theiner, Allen, & Goldstone, 2010) prove that even very simple cooperating agents at the level of the system in which they operate could have an increased intelligence.

There are very few designed metrics that could effectively measure the systems' intelligence. Each of such developed metrics is based on some considerations related to the intelligence, based on different approaches for obtaining

* Corresponding author.
*E-mail address:* ibarna@science.upm.ro (L.B. Iantovics).

http://dx.doi.org/10.1016/j.cogsys.2017.04.006
1389-0417/

the measure of machine intelligence. Winklerová (2013) proposes the collective intelligence of the particle swarm system, assessing it according to some kind of proposed Maturity Model. Anthon and Jannett (2007) consider the agent-based systems intelligence based on the ability to compare alternatives with different complexity. Fox, Beveridge, and Glasspool (2003) designed a benchmark cognitive BDI agent (with Belief-Desire-Intention architecture) model that can be used for comparing agents intelligence. Chmait, Dowe, Green, Li, and Insa-Cabrera (2015) proposed a metric considered universal, appropriate to make an empirical intelligence measure of different agents. Hernández-Orallo and Dowe (2010) proposes the idea of a general test called universal anytime intelligence test, which should be able to measure the intelligence level (any low or any high) of any biological or artificial system. Hernández-Orallo, Dowe, and Hernández-Lloreda (2014) studied the development of universal metrics for measuring the capabilities of cognitive systems, concluding that such metrics should be inspired by the psychometrics used for measuring the human cognitive capacities. Schreiner (2000) presents a theoretical study realized by the US National Institute of Standards and Technology (NIST) related to creating standard measures for intelligent systems, outlining the question on how precisely intelligent systems are defined and how to measure and compare the capabilities that intelligent systems should provide. Hibbard (2011) proposes a measure of intelligence based on a hierarchy of sets of increasingly difficult environments; based on this approach an agent's intelligence is measured according to the ordinal of the most difficult set of environments that may occur. Chmait et al. (2015) present a study related to the intelligence of multiagent systems, focused on the research question related to influencing the intelligence by the communication and observation abilities of the member agents.

Elaborated metrics presented in the scientific literature do not take into account in the intelligence evaluation aspects like: variability in intelligence (higher and lower intelligence). We consider that if the variability is not taken into account in a swarm system, in some situations this could result even in the appearance of erroneous evaluations of the intelligence. Different experimental evaluations usually lead to different results. Elaborated metrics presented in the literature do not take into account the statistically extremely high and statistically extremely low intelligent measures, which we call outlier intelligence values. We consider that outlier intelligence values should not always be taken into consideration, based on the fact that they could lead to erroneous machine intelligence quotient evaluations.

In this paper, we propose an accurate and robust mathematically grounded metric called *MetrIntMeas* (*Metric for the Intelligence Measuring*) for measuring the difficult problem-solving intelligence of a swarm of agents that cooperate with each other. The type of measured intelligence should be established by the human evaluator who would like to obtain the intelligence quotient of the swarm called *CIM* (*Central Intelligence Indicator*). A swarm could be a whole cooperative multiagent system or a part of it – a coalition. Some of the member agents of a multiagent system could form a coalition in order to more efficiently solve the problems by joint collaboration. A multiagent system could be formed by more cooperative swarms. We define the notion *CIM* (*Central Intelligence Indicator*) the indicator of a swarm system's central intelligence tendency. For demonstrating the effectiveness of the metric, we conducted an illustrative case study. A swarm system, formed by reactive agents that cooperate based on some simple rules, able to solve the *TSP* (*Traveling Salesman Problem*) was studied. *TSP* is an example of NP-hard problem, formulated in 1930 (see for example Crisan, Pintea, & Palade, 2016). For different solutions offered to this problem's solving see the "TSP world tour", last updated on September 2013 that illustrates the wide effort put in this problem solving.

There are many studies in the scientific literature that describe systems able to evolve (Blazic, Skrjanc, & Matko, 2014; Liang, Hu, & Kasabov, 2015; Lughofer, Cernuda, Kindermann, & Pratama, 2015). In this paper, we give a new definition to intelligent evolving systems, a class of evolving systems. In the realized case study, we proved by using the *MetrIntMeas* metric, that the studied swarm system does not pass an evolutionary step.

The upcoming part of the paper is organized as follows: Section 2 presents details related to the proposed *MetrIntMeas* metric for measuring the swarm intelligence; in Section 2.1 the *MetrIntMeas* metric as an algorithm is presented; Section 2.2 presents the establishment of the reference intelligence value to which the *MetrIntMeas* metric compares a studied swarm system's intelligence; in Section 2.3 the intelligence indicator calculation based on a measure with more intelligence evaluation components is presented. Section 3 presents a case study for validation of the proposed metric. In Section 4 the application of the *MetrIntMeas* for measuring an intelligent swarm system's evolution is presented. Section 5 presents the discussion related to the performed research. In Section 6 the main conclusions of the research are presented.

## 2. A novel metric for measuring a swarm system's intelligence

### 2.1. MetrIntMeas the proposed metric for measuring of swarm intelligence

In the following, we propose a novel mathematically grounded accurate and robust metric called *MetrIntMeas (Metric for the Intelligence Measuring)*. It is appropriate for measuring the problem-solving intelligence of a swarm system. We consider the agent notion based on the definition given by Russell and Novig (1995). Definition: "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environ-

ment through effectors". We call swarm system a coalition of agents (or swarm multiagent system) situated in the same environment able to communicate with each other, which cooperatively solve each undertaken problem.

In our study, the intelligence is considered from the difficult problem-solving point of view (computational intelligence in difficult problem-solving). The type of measured intelligence should be defined by a human evaluator that would like to obtain the intelligence quotient of the swarm.

For example, in the case of a robotic swarm, he/she could consider the intelligence:

- In the collection of objects: A problem that the swarm should solve consists in the collection of a set of distributed objects.
- In the distribution of objects: A problem that the swarm should solve consists in the distribution of a set of objects to distributed sources.

We denote a studied swarm system with $SWM$, $SWM = \{SW_1, SW_2, \ldots, SW_n\}$. $SWM$ is composed of a group of cooperative agents denoted $SW_1$, $SW_2$, …, $SW_n$. $|SWM|$, $|SWM| = n$ denotes the cardinality (number of agents) of $SWM$.

The algorithm *Metric for the Intelligence Measuring* presents our proposed *MetrIntMeas* metric for measuring the computational intelligence of difficult problem-solving. $Probl = \{Pr_1, Pr_2, \ldots, Pr_N\}$ denotes the problems set that is used for the obtaining of the intelligence indicators. $SampleInteligence = \{SI_1, SI_2, \ldots, SI_N\}$ denotes the measured intelligence indicators sample obtained during the intelligence measurements. $SI_1$ denotes the measured intelligence indicator obtained by $Pr_1$ solving; $SI_2$ denotes the measured intelligence indicator obtained by $Pr_2$ solving; …. $SI_N$ denotes the measured intelligence indicator obtained by $Pr_N$ solving. $|SampleInteligence| = |Probl| = N$ denotes the *SampleInteligence* sample size. $Df = N-1$ denotes the degrees of freedom of *SampleInteligence*. *MetrIntMeas* compares the $SWM$ system's intelligence with the given reference intelligence value denoted *RefIntellig*.

In the *MetrIntMeas* algorithm the following notations were used:

- "@" denotes the effectuation of more calculus. For example: "@Apply the One Sample T-test", denotes the application of the calculus specific to the One Sample T-test described in the scientific literature.
- *Decision* contains the decision related to the considered $SWM$ system's intelligence in comparison with *RefIntellig*.
- $CV$ denotes the *Coefficient of Variation*.

*Indic* used in the frame of the algorithm indicates how the $CIM$ (*Central Intelligence Indicator*) is calculated. If the *SampleInteligence* data is normally distributed (sampled from a Gaussian population), then we consider the mean, $Indic =$ "Mean", $CIM = mean(SI_1, SI_2, \ldots, SI_N)$, as the most appropriate. Elsewhere, we consider the median $Indic =$ "Median", $CIM = median(SI_1, SI_2, \ldots, SI_N)$ as the most appropriate. $HE$ is responsible for the establishment of the type of intelligence wished to be evaluated. $HE$ establishes *Probl* the problems set that will be used in the intelligence measurements.

**MetrIntMeas: Algorithm Metric for the Intelligence Measuring**
**IN:** *RefIntellig; SWM*
**Out:** *N; SampleInteligence*=$\{SI_1, SI_2, \ldots, SI_N\}$; *Indic; Decision*
**Step 1.** *Determination of the sample size*
@Calculate $N$ as the necessary sample size of *Probl*.
@$HE$ establishes $Probl = \{Pr_1, Pr_2, \ldots, Pr_N\}$.
**Step 2.** *Calculation of the intelligence indicators.*
@Calculates the *SampleIntelligence* by evaluating the *Probl* solving intelligence.
**Step 3.** *Statistical analysis of the obtained intelligence indicators.*
@Makes a statistical characterization of the *SampleInteligence*.
@Calculates the $CV$ of the *SampleInteligence*.
**Step4.** *Hybrid decision for the elimination of the outlier intelligence value(s).*
*Decision_het*:="*NO*".
***If*** ($CV > 30$) ***then***
//is requested to $HE$ to take a decision related with the OIVs elimination.
@$HE$ elaborates the *Decision_het.*
***EndIf***
***If*** *(Decision_het='YES')* ***then***
@Eliminate the OIVs using an outlier's detection test.
***EndIf***
**Step5.** *Verification of the SampleInteligence data normality.*
@Verify if *SampleInteligence* is normally distributed.
**Step6.** *Application of a statistical test for the verification of equality*
@Formulate *H0I* (Null Hypothesis of the Intelligence).
@Formulate *HAI* (Alternative Hypothesis of the Intelligence).
***If*** (*SampleInteligence* data is normally distributed) ***then***
**Begin**
*Indic:*="Mean".
@Calculates the $CIM$ as the mean of *SampleInteligence* data.
@Apply the "One Sample T-test".
@Obtain the *P-value* as the "One Sample T-test" result.
**End**
**Else**
**Begin**
*Indic:*="Median".
@Calculates the $CIM$ as the median of *SampleInteligence* data.

@Apply the "Wilkoxon Rank Sum test".
@Obtain the *P-value* as the "Wilkoxon Rank Sum test" result.
**End**
***EndIf***
**Step 7.** *Interpretation of the intelligence evaluation results*
***If*** (*P-value*>α_*metr*) ***then***
**Begin**
//A differentiation in intelligence cannot be realized.
@Accept *H0I.*
@*SWM* intelligence is statistically equal with the reference *RefIntellig.*
**End**
**else**
**Begin**
//Reject *H0I* - a differentiation in intelligence can be realized.
@Accept *HAI.*
***If*** (*CIM* < *RefIntellig*) ***then***
@"*SWM* is less intelligent than the reference intelligence *RefIntellig.*"
***Else***
@"*SWM* is more intelligent than the reference intelligence *RefIntellig.*"
***EndIf***
**End**
***EndIf***
@Presents the calculated *CIM*
**EndMetricIntelligenceMeasuring**

For the verification of the normality different statistical tests can be applied. The most frequently used are the (Razali & Wah 2011): Kolmogorov–Smirnov test (Chakravarti, Laha, & Roy, 1967; Lilliefors, 1967, 1969); Shapiro–Wilk test (Shapiro & Wilk, 1965); Lilliefors test – based on the Kolmogorov–Smirnov test (Dallal & Wilkinson, 1986; Lilliefors, 1967; Lilliefors, 1969) and Anderson–Darling test (Anderson & Darling, 1952; Stephens, 1974, 1986). The Two-Sample Kolmogorov–Smirnov test is used for comparison if two data sets are sampled from the same distribution. The One Sample Kolmogorov–Smirnov Goodness-of-Fit test is applied for the verification of the data normality. In our study, we refer the One Sample Kolmogorov–Smirnov Goodness-of-Fit test.

Razali and Wah (2011) presented a comparative analysis of the Kolmogorov–Smirnov, Shapiro–Wilk, Lilliefors, and Anderson–Darling tests. The conclusion of the study was that Shapiro–Wilk has the best power for a given significance, followed closely by Anderson–Darling.

Stephens (1974) proved that Kolmogorov–Smirnov Goodness-of-Fit One Sample test is less powerful for testing normality than the Shapiro–Wilk test or the Anderson–Darling test, but he outlined that it has some disadvantages. One of the disadvantages of the Shapiro–Wilk test consists in the fact that it does not work well with many identical values.

We propose for the normality testing the application of the both tests, the Kolmogorov–Smirnov Goodness-of-Fit One Sample test, and the Shapiro–Wilk test. This proposal has as rational the maintaining of the generality of the *MetrIntMeas* metric.

We call Null Hypothesis of the Intelligence denoted in the algorithm as *H0I*, the statement that the sample *CIM* is equal from the statistical point of view with *RefIntellig.* This conclusion can be formulated; *SWM* intelligence is equal from the statistical point of view with *RefIntellig*. We call Alternative Hypothesis of the Intelligence and denote it with *HAI* the hypothesis that the *CIM* is different from the statistical point of view from *RefIntellig.* This conclusion can be formulated; *SWM* intelligence is different from the statistical point of view from the *RefIntellig*. The testing of *H0I* and *HAI* should be realized with a significance level denoted α_*metr*.

The *Step1* of the *MetrIntMeas* algorithm indicates the establishment of the sample size *N* (|*SampleInteligence*| = |*Probl*| = *N*) based on a mathematical a priory calculus. The calculus includes: the number of *Tails* (could be 1 or 2); *Effect size* (denoted *D*); α_*metr* (probability to make a type I error); β_*metr* (probability to make a type II error); $TP\ (test\ power) = 1-\beta\_metr$. We considered as many times the most appropriate: using the test with two tails; $\alpha\_metr = 0.05$; $\beta\_metr = 0.2$; $TP = 0.8$. The output of the calculus was the *Df* (|*SampleInteligence*|−1) and δ (the Noncentrality Parameter). A central distribution describes how a test statistic is distributed when the tested difference is null. The noncentral distributions describe the distribution of a test statistic when the null is false. A noncentrality parameter is families of probability distributions that are related to other families of distributions ("central" families of distributions) (Dodge, 2003).

α_*metr* represents the probability of rejecting *H0I* when it is true. α_*metr* is a parameter of the algorithm. We proposed for α_*metr*, in most of the cases the value $\alpha\_metr = 0.05$. β_*metr* denotes the probability to make a type II error. A type II error is the failure to reject a false null hypothesis (incorrect failure to reject a false null hypothesis). We denote with *TP* the test power. We propose as many times the most appropriate $\beta\_metr = 0.2$, $TP = 0.8$. We suggested these values for α_*metr* and β_*metr* based on the type I and type II errors and the relation between them. A type I error is detecting an effect that is not present, while a type II error is failing to detect an effect that is present. *D* (the *effect size*) is another important statistical indicator, which allows the quantitative measurement of the strength of a phenomenon (Kelley & Preacher, 2012). Cohen gives as a guideline for the effect size the following values (Cohen, 1992): *Small effect size* between 0.2 and 0.3; *Medium effect size* around 0.5; *Large effect size* for 0.8 or higher value.

The sample size is finally established based on the sample size calculation result and taking into consideration the cost of the experimental measurements of the intelligence (1). For example, we consider a swarm of robotic agents

that operate in a physical environment by searching for objects. The realization of experimental intelligence measurements has a cost, related to: time for realization of the measurements and/or resource consumption, and some others.

$$Cost(SampleIntelligence) = Cost(SI_1) + Cost(SI_2) + ..... + Cost(SI_N) \quad (1)$$

During the statistical analysis of the intelligence, as first step a statistical characterization should be realized (Mann, 1995; Nick, 2007) by calculating for the *SampleIntelligence* data the values for the most important statistical indicators. *Mean* ($Mean = (SI_1 + SI_2 + \ldots + SI_N)/N$); *Standard Deviation* (*SD*); *Variance* ($Variance = SD^2$); *SE* (Standard Error of the Mean), $SE = SD/sqrt(N)$, where *sqrt* denotes the square root; *Median* (is the "middle" of a sorted list of numbers); *Mode* (the most frequent value); *Kurtosis* (Kurtosis is a measure of whether the data are heavy-tailed or light-tailed relative to a normal distribution); *Skewness* (is a measure of the lack of symmetry); *Minimum*(the smallest value); *Maximum*(the highest value); and *CL* (*Confidence Level*), *Lower CI*, *Upper CI*. In most of the cases, we propose the choosing of the 95.0% confidence, $CL = 95\%$. *CV* (*Coefficient of Variation*), $CV = 100 \times (SD/Mean)$.

*CV* indicates the homogeneity-heterogeneity of the *SampleInteligence*. We consider the data classification based on the variability, as was proposed by Marusteri and Bacarea (2010). $CV \in [0,10)$ indicates homogeneous data; $CV \in [10,30)$ indicates relatively homogeneous data; $CV \geq 30$ indicates heterogeneous data. The statistical characterization could be appropriate for an evaluator to formulate some additional characterization of a system's intelligent behavior. For example, in the case of homogeneity (decision that could be taken based on the *CV* value), it is expected that the system does not present high variations in intelligence.

Step 4 of the algorithm indicates that if the data is heterogeneous, then a hybrid decision is realized (a combined human-computing system decision) for approving or declining on the application of an outlier's detection test. If *SampleInteligence* is heterogeneous *HE* is asked to decide if he/she agrees or not with the outlier intelligence values elimination. The rationale for this hybridization is based on the fact that *HE* (being a human specialist) detains some problem and system specific knowledge that allows a better decision taking than an automatic decision.

There are many tests described in the scientific literature, for the outliers detection like: Peirce's criterion (Stigler, 1978), Grubbs test (Barnett & Lewis, 1994; Grubbs, 1950), Dixon's Q test (Dean & Dixon, 1951) and Chauvenet's criterion (Ross, 2003; Zerbet & Nikulin, 2003). We chose for our research the Grubbs test for outliers detection applied with significance level $\alpha\_out = 0{,}05$. At an application, the Grubbs test is able to detect a single outlier. Based on this fact, when applying the test, if an outlier value is detected, then the outliers detection test could be applied again. At a new application, it is possible to detect other outliers. This test could be repeated recursively until no outlier value is detected.

For the *H0I* testing, we consider as the most appropriate (see the Step 6 of the algorithm) the application of the *One Sample T-test* (Marusteri & Bacarea, 2010) in the case of normally distributed data. For the *H0I* testing, we consider as the most appropriate (see the Step 6 of the algorithm) the application of *Wilkoxon Rank Sum test* (Marusteri & Bacarea, 2010) in case of data that is not sampled from a Gaussian distribution. The "One Sample T-test" and the "Wilkoxon Rank Sum test" allow the examination of the *CIM* difference from the *RefIntellig*.

Finally, as output, the *Algorithm Metric for the Intelligence Measuring* calculates the *SWM* system's *CIM*, and establishes if this is statistically equal, lower or higher with the initially given *RefIntellig* reference intelligence indicator. The proposed metric also allows the classification. In case of a given reference intelligence value, it establishes if the considered swarm system belongs or not to the same intelligence class.

### 2.2. Establishment of the reference intelligence value

In the previous section, we presented the *MetrIntMeas* metric for the comparison of a swarm system's intelligence with a specified reference intelligence indicator, denoted *RefIntellig* (if the studied swarm system belongs to the intelligence class indicated by the *RefIntellig*). Now, we present the algorithm called *Reference Intelligence Indicator Establishment* for the determination of the *RefIntellig* value. In the algorithm the following notations are used: *SWMB* denotes the considered swarm system used in order to calculate *RefIntellig*; *M* denotes the sample size of the intelligence indicators; $ProblB = \{Pr_1, Pr_2, \ldots, Pr_M\}$ denotes the problems used in the intelligence indicators measuring; $ExprimReference = \{SI_1, SI_2, \ldots, SI_N\}$ the intelligence indicators obtained as result of the *ProblB* solving intelligence measurements.

**Algorithm Reference Intelligence Indicator Establishment**
**IN:** $SWMB;\ ProblB=\{Pr_1, Pr_2, \ldots, Pr_M\}$
**OUT:** $ExprimReference=\{SI_1, SI_2, \ldots\ldots, SI_M\}$;
*RefIntellig; Indic*
**Step1.** *Calculation of the intelligence indicators values.*
@Calculates the *ExprimReference* indicators.
**Step 2.** *Statistical analysis of the obtained intelligence indicators.*
@Calculates the *CV* of *ExprimReference* data.
**Step3.** *Hybrid decision for the elimination of the outlier intelligence value(s).*
*Decision_het*:="*NO*"
***If*** $(CV > 30)$ ***then***
//*HE* takes a decision related to the OIVs elimination.
@*HE* decision *Decision_het.*
***EndIf***

***If*** *(Decision_het='YES')* ***then***
**@**Eliminate the outlier intelligence values using an outlier's detection test.
***EndIf***
**Step 4.** *Verification of ExprimReference data normality.*
@Verify if *ExprimReference* is normally distributed.
***If*** (*ExprimReference* data is normally distributed) ***then***
**Begin**
*Indic:*="Mean".
@Calculates the *RefIntellig* as the mean of *ExprimReference* data.
**End**
**Else**
**Begin**
*Indic:*="Median"
@Calculates the *RefIntellig* as the median of *ExprimReference* data.
**End**
***EndIf***
**Step 5.** *Presentation of the obtained result.*
@Presents the calculated *RefIntellig.*
**EndReferenceInteligenceIndicatorEstablishment**

### 2.3. Intelligence indicator based on more intelligence evaluation components measure

The previous sections argued that our metric is appropriate for swarm systems, where an intelligence indicator can be expressed as a single value. If necessary (could be the case of more complex swarm systems), this intelligence indicator can be calculated as a weighted sum of more elementary (could be or not atomic) intelligence evaluation components measure. The intelligence evaluation components and their weights should be identified by the human evaluator. An intelligence evaluation component is an elementary factor that is considered to contribute to a problem-solving intelligence evaluation measure. As examples of frequent intelligence evaluation components, we mention: the problem-solving time, the obtained solution accuracy, the capacity of handling different types of uncertainties (erroneous or missing data) and some others. Eq. (2) presents the general case when an intelligence indicator denoted with *SI* is calculated as the weighted sum of *r* intelligence evaluation components measure. The intelligence components being denoted with 1, 2,..., *r*.

Just for illustrative purposes, we present as an example an intelligent robotic swarm specialized in collecting and distributing objects in a physical environment. A type of problem that must be solved by the swarm consists of:

- *TypeA-problem:* collection of a set of distributed objects in the environment;
- *TypeB-problem:* distribution of a set of objects in the environment to different places.

The intelligence evaluation for a *TypeA-problem* problem-solving consists in the evaluation of the intelligence in the collection of a set of distributed objects. The object collection intelligence could include as measurable intelligence evaluation components:

Comp1) percent (%) of the successfully collected objects (the collection of some objects could be missed).

Comp2) the duration of objects collection (until all the required objects are collected or the collection is interrupted).

Comp3) the amount of power consumption at the swarm level. Power consumption is important in order to maintain the autonomy in operation of the swarm.

Comp4) new information or knowledge learned by the swarm during the problem solving. Learned information or knowledge could help the swarm in the upcoming problems solving.

Comp5) measured efficiency in the most frequent obstacles avoidance. Can be calculated as the average time of the avoidance for example.

$$SI = wg_1 \mathrm{x}\, ic_1 + wg_2 \mathrm{x}\, ic_2 + \ldots + wg_\mathrm{q} \mathrm{x}\, ic_\mathrm{r};$$
$$wg_1 + wg_2 + \ldots.. + wg_\mathrm{r} = 1 \quad (2)$$

where: $ic_1$, $ic_2$,...,$ic_r$ represent the considered intelligence evaluation components measure at a problem-solving; $wg_1$, $wg_2$,...,$wg_r$ represent the weights of the intelligence evaluation components ($wg_1$ corresponds to component 1; $wg_2$ corresponds to component 2; .... ; $wg_r$ corresponds to component *r*). The weights indicate the importance of the intelligence components in the establishment of the problem-solving intelligence. $wg_k = wg_j$ means that the components *k* and *j* have the same importance in the quantitative value of the intelligence indicator. $wg_i < wg_j$ means that the component numbered *i* is less important than the component *j* in establishing of the intelligence.

## 3. A case study for measuring a swarm system's intelligence using MetrIntMeas

In order to prove the effectiveness of the proposed *MetrIntMeas* metric, we realized a case study. The *TSP* (Traveling Salesman Problem) solving was considered, a well known *NP*-hard problem (Crisan et al., 2016; Leeuwen, 1998). The *TSP* can be defined as follows: a map of cities is given as well as the distances between each pair of cities. The problem statement is what is the shortest possible route that visits each city exactly once and returns to the origin city? Given *k* as the number of cities to be visited, the total number of possible routes covering all cities is given as *(k-1)!/2*.

*TSP* is an intensively studied problem, and it has many applications, like: drilling of printed circuit boards (Grötschel, Jünger, & Reinelt, 1991); overhauling gas turbine engines (Plante, Lowe, & Chandrasekaran, 1987); computer wiring (Lenstra & Kan, 1974). The elaboration

of more efficient solving is open, based on the fact that it is an *NP*-hard problem.

In the case study, we considered swarm systems composed of very simple cooperating mobile agents that mimic the operation of a natural ant colony. The agents communicate via signals. It is similar to the communication of natural ants by pheromones. A communicated signal is not directed to a specific agent. All the agents that reach the signal will take it into consideration in the decision related to the next place that they will visit. Even if the cooperation is based on a simple rule, it makes the swarm system robust and also scalable.

We made experimentations for different swarm systems: *Elitist Ant System*, *Ranked Ant System*, *Best-Worst Ant System*, *Min–Max Ant System* and *Ant Colony System*. Finally, for the case study, we chose the swarm system that operated as an *Elitist Ant System* (Dorigo et al., 1996). The operation of an *Elitist Ant System* in the Section 3.3 is presented. Swarm system that operates as *Elitist Ant Systems* has a wide range of applications like the: Post-Enrolment Course Timetabling Problem (Jaradat & Ayob, 2010) and TSP solving (Martinovic & Bajer, 2012). The swarm system's intelligence is analyzed for the solving of *TSP* with 40 nodes ($dim = 40$, represents the problem's dimensionality) in a graph. For the determination of *RefIntellig,* we considered a swarm system that operated as an *Ant System* (Colorni, Dorigo, & Maniezzo, 1991; Dorigo, 1992) in solving the *TSP*. The basic idea of an *Ant System* is presented in Section 3.1. We took this decision based on the fact that swarm systems that operate as *Ant System* are the most frequent. They represented the inspiration of many other swarm systems design.

### 3.1. A swarm system that operates as an ant system considered

Marco Dorigo proposed first in 1992 in his Ph.D. thesis (Colorni et al., 1991; Dorigo, 1992), the bio-inspired problem-solving inspired from the natural ants, by mimicking how they search for food. In the following, we briefly explain the idea of a swarm system that operates as an *Ant System*. Initially, each agent is placed on some chosen city (node of the graph). An agent $agent_k$ currently at nod (city) $city_i$ chooses to move to city $city_j$ by applying the following probabilistic transition rule:

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha}\,[\eta_{ij}]^{\beta}}{\sum_{l \in J_k(i)} [\tau_{il}(t)]^{\alpha}\,[\eta_{il}]^{\beta}} & \text{if } j \in J_k(i) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

In formula (3) the following notations are used: $J_k(i)$ is a set of cities that can be visited when the agent is at $city_i$; $\eta_{ij}$ associated to the edge $(i,j)$, represents the heuristic visibility of edge $(i, j)$, many times $\eta_{ij} = 1/d_{ij}$; $d_{ij}$ represents the distance between city $city_i$ and city $city_j$; $\alpha \geq 0$ is an adjustable parameter that controls the relative weights of the pheromone trail; $\beta \geq 0$ is an adjustable parameter of the heuristic visibility. When $\alpha = 0$, the closed vertex is more likely to be selected. When $\beta = 0$, only pheromone amplification is at work: this will lead the system to a stagnation situation (all the agents generate a sub-optimal tour). Based on these aspects a trade-off between edge length and pheromone intensity should be realized.

In the case of an *Ant System* the pheromone amount on each path will be adjusted according to formulas (4). This adjustment is realized after each agent completes its tour.

$$\begin{aligned} &\tau_{ij}(t+1) = (1-\rho)\,\tau_{ij}(t) + \Delta\tau_{ij}(t) \\ &\Delta\tau_{ij}(t) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t) \\ &\Delta\tau_{ij}^{k}(t) = \begin{cases} Q/K_k, & \text{if } (i,j) \in \text{ tour done by agent } k \\ 0 & \text{otherwise} \end{cases} \end{aligned} \quad (4)$$

In formulas (4) the following notations were used: $Q$ is an arbitrary constant (many times its value should be set to 1); $L_k$ is the length of the tour performed by the $agent_k$; $1-\rho$ represents the pheromone decay parameter, where $0 < \rho < 1$, it represents the trail evaporation when the agent chooses a city and decides to move; $m$ represents the number of agents.

### 3.2. Establishment of the reference intelligence value for the case study

As a first step, we established the reference intelligence value denoted *RefIntellig* using the *Reference Intelligence Indicator Establishment* algorithm. In the quality of human evaluator, we considered as the intelligence indicator the *best to date* obtained solution that is obtained at the end of a problem's solving by the swarm.

The intelligence evaluations for establishing *RefIntellig* were realized. The parameters settings of the swarm operation, established empirically, were: $M = 25$ experimental simulations; *generated epoch* $= 1000$; *cities extent* $= 9.8$; $\alpha = 1$; $\beta = 1$; *evaporation* $= 0.1$; 40 cities considered; swarm system formed by 10 agents. Table 1 presents the results of the $M$ intelligence evaluations. The cities were placed randomly on the map at each problem-solving intelligence measuring. The distance between cities was considered as the Euclidian distance, based on the cities placement on the map.

Table 2 present the results of the statistical characterization realized based on the *ExprimReference* data. The obtained $CV$ value, $CV \approx 3.59$, $CV < 10$, indicates that *ExprimReference* is homogeneous. Implicitly an outliers detection test will not be applied (a hybrid decision is not necessary).

Table 1
The obtained *ExprimReference* values.

| | | | | |
|---|---|---|---|---|
| 51.41 | 49.79 | 54.76 | 52.41 | 49.37 |
| 49.65 | 50.8 | 53.27 | 55.75 | 53 |
| 51.91 | 49.23 | 53.43 | 51.53 | 54.64 |
| 53.13 | 52.82 | 50.06 | 52.89 | 52.59 |
| 54.7 | 52.43 | 55.59 | 52.66 | 51.57 |

Table 2
Results of the statistical characterization realized on the *ExprimReference* data.

| | |
|---|---|
| Mean | 52.3756 |
| Standard error | 0.3759 |
| Median | 52.59 |
| Standard deviation | 1.88 |
| Variance | 3.533284 |
| Kurtosis | −0.65888 |
| Skewness | −0.01424 |
| Range | 6.52 |
| Minimum/maximum | 49.23/55.75 |
| CL/[LowerCI, UpperCI] | 95%[51.6, 53.152] |
| CV | 3.58889 |

Table 3
Results of the Kolmogorov-Smirnov test for *ExprimReference* data.

| | |
|---|---|
| KS | 0.1073 |
| *P*-value | ≈0.1 |
| Passed normality test | Yes |

Table 3 present the results of the One Sample Kolmogorov–Smirnov Goodness-of-Fit test for the verification of *ExprimReference* data normality. We considered the application of the test with the significance level $\alpha\_kolmog$, $\alpha\_kolmog = 0.05$. As a test result, the *P-value* $\approx 0.1$ and *KS* (*Kolmogorov–Smirnov distance*) $= 0.1073$ were obtained. The obtained result, *P-value* $> \alpha\_kolmog$ indicates that the data normality passed. Based on this fact, *RefIntellig* (the *CIM*) is calculated as the mean of *ExprimReference* data, resulting in *RefIntellig* $= 52.3756$.

### 3.3. The considered swarm system based on the idea of elitist ant system

In Section 3.1, swarm systems that operate as Ant Systems were presented. The solution construction and evaporation are defined by the formulas (4). In case of an Elitist Ant Algorithm (Dorigo et al., 1996), the pheromone update is modified as follows (5): at each iteration, the best to date agent deposits an additional quantity of pheromone on paths it traveled:

$$\tau_{ij} = \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e \cdot \Delta\tau_{ij}^{bs} \quad (5)$$

where: $e$ should be specified at the beginning of algorithm running (it is a parameter of the algorithm); $T^{bs}$ is the best to date round trip; $\tau_{ij}^{bs} = Q/L^{bs}$ if the path $ij$ is from $T^{bs}$. In some studies, it is indicated that $e$ should take a value between 4 and 6. The elitist agent keeps depositing an additional pheromone on the same paths until the best to date solution is changed.

### 3.4. Measuring the intelligence of the studied swarm system

For the intelligence measuring using the proposed metric, we considered a swarm system that operated as an *Elitist Ant System* in solving the TSP. For the determination of the *SampleInteligence* sample size, we performed the computation mentioned in Section 2.1, based on the following values: $Tails = 2$; $Effect\ Size(d) = 0.413$; $\alpha\_metr = 0.05$; $\beta\_metr = 0.2$, $TP(TP = 1-\beta\_metr)=0.8$. The calculus results were: $N$ (*Total sample size*) $= 48$, $Df = N-1 = 47$, *Critical* $t \approx 2.0117$, *Noncentrality parameter* $\delta = 2.8613$.

For obtaining the *SampleInteligence* data set, we realized intelligence measurements, with results presented in Table 4. The parameters of the evaluated *Elitist Ant System* where: a map formed by 40 cities; *generated epoch* $= 1000$; *cities extent* $= 9.8$; $\alpha$ *(of the swarm system algorithm see Section 4.1)* $= 1$; $\beta$ *(of the swarm system algorithm see Section 4.1)* $= 1$; *evaporation* $= 0.1$; $N = 48$-experimental simulations. Fig. 1 represents the intelligence variability graphically.

Table 5 presents the results of the statistical characterization performed on *SampleInteligence* data. The obtained $CV$ value, $CV \approx 6.36$, $CV < 10$ indicates that the data is homogeneous. This having as meaning, the hybrid decision is not necessary. Implicitly an outliers' detection test will not be applied.

We chose to apply more normality tests (more and less powerful tests). Based on the fact that even the most powerful Shapiro–Wilk test was used, we chose as the significance level $\alpha\_norm = 0.045$. Table 6 presents the results of the normality test by using: the One Sample Kolmogorov–Smirnov Goodness of Fit Test of Normality (Chakravarti et al., 1967; Lilliefors, 1967, 1969); Lilliefors – based on the Kolmogorov–Smirnov test (Dallal & Wilkinson 1986; Lilliefors, 1967, 1969) and the Shapiro–Wilk test (Shapiro & Wilk, 1965). All of the tests results indicate that the intelligence indicator's data is normally distributed.

Based on the fact that the normality test passed, and the $CV$ indicates homogeneous data, it can be deduced according to the *MetrIntMeas* algorithm that *CIM* should be calculated as the mean of *SampleInteligence*, resulting in $CIM = 51.83813$, with: $LowerCI = 50.881$ and $UpperCI = 52.796$. $CL = 95.0\%$ was considered as the confidence level.

Table 4
The obtained *SampleInteligence* data for the studied *Elitist ant system.*

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 52.05 | 51.76 | 57.63 | 51.27 | 51.62 | 52.57 | 50.25 | 51.82 | 54.31 | 52.88 | 49.58 | 44.82 |
| 54.57 | 49.68 | 49.46 | 56.72 | 48.62 | 51.82 | 51.98 | 51.87 | 46.28 | 48.26 | 47.62 | 54.53 |
| 55.57 | 51.92 | 52.17 | 51.52 | 56.43 | 49.75 | 53.76 | 51.74 | 55.2 | 48.83 | 55.08 | 49.99 |
| 59.38* | 48.91 | 51.47 | 57.27 | 45.57 | 51.75 | 57.51 | 51.81 | 52.19 | 53.32 | 49.54 | 45.58 |

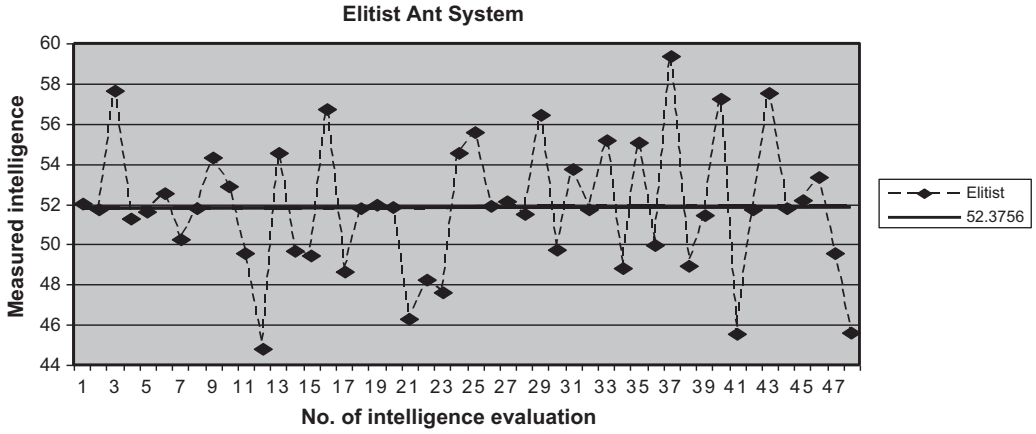



Fig. 1. Graphical representation of the intelligence variability.

Table 5
Statistical characterization of the *SampleInteligence* data.

| | |
|---|---|
| Mean | 51.83813 |
| Standard error | 0.4756 |
| Median | 51.815 |
| Mode | 51.82 |
| Standard deviation | 3.295 |
| Sample variance | 10.85518 |
| Kurtosis | −0.07847 |
| Skewness | 0.054674 |
| Range | 14.56 |
| Minimum/maximum | 44.82/59.38 |
| CL/[LowerCI, UpperCI] | 95%/[50.881, 52.796] |
| CV | 6.355785 |

For the statistical comparison of the sample *SampleInteligence* with the *RefIntellig,* the parametric *One Sample T-test* was applied. See Step 6 of the *Metric for the Intelligence Measuring* algorithm, for the decision of choosing of the right statistic test. The test with two tails, *Df* (degrees of freedom) = 47, was applied. The obtained test results were the: *P-value* = 0.2641 and *Critical T* = 1.130. Based on the fact that the *P-value* > α_*metr*, it can be concluded that *H0I* can be accepted. There is no statistical difference between the intelligence of the *Elitist Ant System* and *RefIntellig.* Based on this fact, the studied swarm system (*Elitist Ant System*) could be considered belonging from the intelligence point of view to the class of all swarm systems whose intelligence is equal to the reference intelligence value *RefIntellig.*

In order to answer the research question "if OIVs were obtained", we applied the Grubbs test on the *SampleInteligence* data. It identified a single "suspicious" intelligence indicator value (marked in Table 4 with *). A "suspicious" intelligence value is different from the others but it cannot be considered as an *OIV*.

## 4. Application of the *MetrIntMeas* for measuring an intelligent swarm system's evolution

In this section, we present the applicability of *MetrIntMeas* metric for measuring an intelligent swarm system's evolution from the intelligence point of view. We give a definition to the intelligent evolving systems.

### 4.1. Evolving systems

The embodied systems are able to sense the environment using sensors and execute actions in the environment using effectors. There is no unanimous definition in the scientific literature of the ES (Evolving systems). Generally, the evolving systems are considered as those embodied systems that operate in a dynamically changing complex environment. Evolving systems many times evolve gradually based on methods like: autonomous learning, inheritance, self-adaptation, or changing of the structure. There are different types of evolving systems described in the scientific literature (Blazic et al., 2014; Liang et al., 2015; Lughofer et al., 2015). There are almost inexistent developed metrics that are able to make an accurate measure of evolution. We consider that such metrics are mandatory because they can realistically verify/prove if/that a system has evolved.

Evolutionary Computation (Holland, 1975; Koza, 1992; Yao, 1999) has applications for various problem-solving

Table 6
Results of the normality tests for the *SampleInteligence* data.

| | One sample Kolmogorov-Smirnov test | Kolmogorov-Smirnov test with Lilliefors significance correction | Shapiro-Wilk test |
|---|---|---|---|
| Statistic | 0.1241 | 0.124 | 0.977 |
| *P*-value (of the normality test) | 0.0618 | 0.062 | 0.460 |
| Normality test passed | *P*-value > α_norm/"Yes" | *P*-value > α_norm/"Yes" | *P*-value > α_norm/"Yes" |

like: semantic aware evolutionary search (Kattan & Ong, 2015); maximum lifetime routing and energy efficiency in sensor mesh networks (Rahat, Everson, & Fieldsend, 2015); multimodal optimization (Stoean, Preuss, Stoean, & Dumitrescu, 2010); optimization based on the Endocrine Control Evolutionary Algorithm (Rotar, 2014); classification based on cooperative co-evolution (Stoean & Stoean, 2009); Stoean, Preuss, Stoean, El-Darzi, and Dumitrescu (2009) propose an evolutionary techniques as an alternative to machine learning based on Support Vector Machine; the evolutionary reorganization of multiagent systems structure (Iantovics, & Zamfirescu, 2013).

Some evolving systems used different ways of evolutionary algorithms in order to evolve. Iantovics and Zamfirescu (2013) presented an intelligent evolving multiagent system called *ERMS* (*Extended Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity*). *ERMS* used an evolutionary learning algorithm that allows the learning of some rules for the system's reorganization. The learned rules are retained in a rule-base. Using the learned rules, the system is able to intelligently reorganize its structure based on the pattern that respects the problems received for solving.

### 4.2. Measuring an intelligent system's evolution

We consider that the intelligence is not a mandatory property of an evolving system. Henceforth, we define the intelligent evolving systems as follows.

**Definition of an intelligent evolving system**

*We call an intelligent evolving system an embodied intelligent artificial system able to make one or more evolutionary steps during its life cycle. We call evolutionary step a measurable increase in the computational intelligence in difficult problem-solving. It should be measured by using the MetrIntMeas accurate and robust metric. MetrIntMeas must indicate a statistically significant difference (increase) between the measured intelligence before making the evolutionary step and after it was made.*

We chosen the specific intelligent swarm systems used as case study presented in Section 3, by taking into consideration the study of the evolution in intelligence. Initially considered swarm system, operated as an Ant System and the modification in the agents' behavior having as the effect the operation as an Elitist Ant System. This can be even the result of simple rote learning, in that the rule, which defines the behavior of the agents, is simply copied from another swarm without using any sophisticated learning technique. The reference intelligence indicator *RefIntellig* was calculated for the Ant System. After then, by applying the *MetrIntMeas* metric, it was measured if the Elitist Ant System intelligence is equal or different from *RefIntellig*. The obtained result was that there is no difference in intelligence. Based on this result, it can be concluded that the swarm system has not evolved. It does not pass an evolutionary step in intelligence.

## 5. Discussions

We observed that in many intelligent swarm systems (Mamei et al., 2003; Stradner et al., 2013), cooperative multiagent systems generally (Johnson et al., 2013; Iantovics and Zamfirescu, 2013; Brady et al., 2014), an agent could not detain a high intelligence by itself for a very efficient operation and problem-solving. Each of the constituting agents should have the capacity to communicate and cooperate with other agents. An agent could communicate information/data that could help one or more other agents in increasing their efficiency in operation. An efficient and flexible cooperation could have as result a more efficient problem-solving. A conclusion of our researches and some other researches described in the scientific literature (Mamei et al., 2003; Iantovics and Zamfirescu, 2013; Stradner et al., 2013) consists in the fact, that even if the individual agent's intelligence is very limited, an increased global intelligence at the system's level emerges in the swarm system. The intelligence emergence of the swarm is the result of the efficient and flexible cooperation during the problem-solving. It can be considered that there is some kind of cognition at the swarm level. Langley, Laird, and Rogers (2009) examine the main motivations for research on cognitive architectures. Some cognitive systems architecture described in the scientific literature were reviewed.

The intelligence of a swarm is not the sum of the intelligence of the agents' members. Our proposed *MetrIntMeas (Metric for the Intelligence Measuring)* metric purpose is to measure this intelligence at the level of the swarm (to measure the swarm intelligence). In our approach, we considered the computational intelligence of the swarm system in difficult problem-solving.

*MetrIntMeas* allows the classification of the swarm systems based on their intelligence. The metric is useful for choosing/opting between swarm systems based on their problem-solving intelligence. Based on an established reference intelligence indicator value, the design of a swarm system could be requested that has that reference intelligence level or it is more intelligent.

The metric analyzes the intelligence indicator data and based on this it applies the most appropriate statistical tests. It treats aspects like the intelligence variability and OIVs (outlier intelligence values). As utility, it must be mentioned that it could be used to detect very high or very low OIVs. This could be important in order to identify problems for whose solving the swarm manifest statistically very low or statistically very high intelligence. The highest intelligence value does not necessary means that it is an *OIV*. The lowest intelligence value does not necessary means that it is a low *OIV*.

The accuracy of the metric is based on the fact that the measuring is statistically grounded. In the case study, by making some intelligence evaluations, intelligence indicators are obtained. Based on these values the *CIM* (central intelligence indicator) is calculated. *CIM* is the calculated intelligence quotient of the swarm. *CIM* indicates the cen-

tral intelligence tendency of the swarm system. Repeating the intelligence measurements on another set of problems slightly different intelligence indicator values will be obtained. Very probably the newly calculated *CIM* will have a slightly different value from the previous one. But, the result of the metric related with the swarm systems intelligence measuring will be the same. The robustness consists in the fact that the sample intelligence indicator data should not be necessarily normally distributed. It could contain outlier values, missing values and even could admit some erroneous indicator values (if they are very different, they could be detected by the outliers' detection test, if they are not very different from the other values, they does not influence the conclusions).

*MetrIntMeas* is appropriate for swarm systems, where an intelligence indicator can be expressed as a single value. Section 2.3 treats the case of swarm systems, where this indicator can be calculated as a weighted sum of some elementary (could be or not atomic) intelligence evaluation components identified by the human evaluator.

We gave a new definition to the intelligent evolving systems. Intelligent evolving systems are intelligent system able to make one or more evolutionary steps during their life cycle. A realized evolutionary step consists in a measurable increase in intelligence by using our proposed *MetrIntMeas* metric. In our definition, it does not count how evolution is realized. It could be the result of an autonomous learning, for example, that is the most usual case. Our definition refers to any intelligent system able to evolve, a subclass of intelligent evolving systems being the intelligent evolving swarm systems.

Evolution measuring has practical utility. In the case of an implemented intelligent system, it can be verified if it evolved in intelligence. For example, we consider a scenario of an intelligent agent able to learn. The agent is endowed with the capacity to learn autonomously during its life cycle. The agent's builder is unable to determine the effect of the learning at the moment of the agent creation. During the agents' life cycle it could be verified if it has evolved in intelligence, as the result of the autonomous learning.

## 6. Conclusions

There are many designed swarm systems (software, robotic and hybrid software-robotic), but the effective intelligence measuring of these systems is still an open research direction. In this paper, we proposed a novel accurate and robust metric called *MetrIntMeas (Metric for the Intelligence Measuring)* for measuring a swarm system's intelligence in solving difficult problems. The type of measured intelligence must be defined by the human evaluator who would like to obtain the intelligence quotient of the swarm. The starting idea of the metric consists in the fact that the intelligence measuring of a swarm system should be based on some problem-solving intelligence evaluations. These measurements result in obtaining a set of intelligence indicators. Based on these intelligence indicators by using the metric, the intelligence level/quotient of the studied swarm system is established. The metric allows the classification of the swarm systems based on their intelligence. For proving the effectiveness of the metric, we realized a case study for a swarm system specialized in solving an NP-hard problem (TSP-Traveling Salesman Problem).

We had given a new definition to intelligent evolving systems. As an application of the *MetrIntMeas* metric, we realized a case study for the metric appropriateness for the verification of whether an intelligent swarm system made an evolutionary step. Based on the obtained results, it can be concluded that the studied swarm system has not evolved from the intelligence point of view.

An important property of the metric that should be mentioned consists in its universality. In our approach, we treated the metric application for cooperative swarm systems (composed of software or robotic agents; could be even hybrid systems) intelligence measuring. But, it could be applied to intelligent systems that operate in isolation (to measure the problem-solving intelligence of an agent for example). There is no restriction for the application of the metric to intelligent systems that solve a specific class(es) of problem(s). Based on a comprehensive study of the scientific literature, we consider that our proposed metric is original and will represent the basis for intelligence quotient/level measuring and measure the evolution of agent-based systems in many future researches.

## Acknowledgment

Laszlo Barna Iantovics acknowledge the support of the COROFLOW project PN-III-P2-2.1-BG-2016-0343.

## References

Akay, B., & Karaboga, D. (2012). A modified artificial bee colony algorithm for real parameter optimization. *Information Sciences, 192*, 120–142.

Anderson, T. W., & Darling, D. A. (1952). Asymptotic theory of certain "goodness-of-fit" criteria based on stochastic processes. *Annals of Mathematical Statistics, 23*, 193–212.

Anthon, A., & Jannett, T. C. (2007). Measuring machine intelligence of an agent-based distributed sensor network system. In K. Elleithy (Ed.), *Advances and innovations in systems, computing sciences and software engineering* (pp. 531–535). Springer.

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3rd ed.). Wiley.

Blazic, S., Skrjanc, I., & Matko, D. (2014). A robust fuzzy adaptive law for evolving control systems. *Evolving Systems, 5*(1), 3–10.

Brady, S. G., Fisher, B. L., Schultz, T. R., & Ward, P. S. (2014). The rise of army ants and their relatives: diversification of specialized predatory doryline ants. *BMC Evolutionary Biology, 14*(93), 1–14.

Chakravarti, I. M., Laha, R. G., & Roy, J. (1967). *Handbook of methods of applied statistics* (Vol. I, pp. 392–394). John Wiley & Sons.

Chmait, N., Dowe, D. L., Green, D. G., & Li, Y. F. (2015a). Observation, communication and intelligence in agent-based systems. In J. Bieger, B. Goertzel, & A. Potapov (Eds.). *Artificial general intelligence* (Vol. 9205, pp. 50–59). Lecture Notes in Computer Science.

Chmait, N., Dowe, D. L., Green, D. G., Li, Y. F., & Insa-Cabrera, J. (2015b). *Measuring universal intelligence in agent-based systems using the anytime intelligence test* Technical Report, Monash University, Report Number, 2015/279. .

Cohen, J. (1992). A power primer. *Psychological Bulletin, 112*(1), 155–159.
Colorni, A., Dorigo, M., & Maniezzo, V. (1991). Distributed optimization by ant colonies. In *Actes de la première conférence européenne sur la vie artificielle, Paris, France* (pp. 134–142). Elsevier.
Crisan, G. C., Pintea, C. M., & Palade, V. (2016). Emergency management using geographic information systems. Application to the first Romanian traveling salesman problem instance. *Knowledge and Information Systems. An International Journal*, 1–21.
Dallal, G. E., & Wilkinson, L. (1986). An analytic approximation to the distribution of Lilliefors's test statistic for normality. *The American Statistician, 40*(4), 294–296.
Dean, R. B., & Dixon, W. J. (1951). Simplified statistics for small numbers of observations. *Analytical Chemistry, 23*(4), 636–638.
Dodge, Y. (2003). *The oxford dictionary of statistical terms*. Oxford University Press.
Dorigo, M. (1992). *Optimization, learning and natural algorithms* PhD thesis. Italy: Politecnico di Milano.
Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B, 26*(1), 29–41.
Feder, T. (2007). Statistical physics is for the birds. *Physics Today, 60*(10), 28–30.
Fox, J., Beveridge, M., & Glasspool, D. (2003). Understanding intelligent agents: analysis and synthesis. *AI Communications, 16*(3), 139–152.
Grosz, B. J., & Hunsberger, L. (2006). The dynamics of intention in collaborative activity, cognition, joint action and collective intentionality. *Cognitive Systems Research, 7*(2–3), 259–272.
Grötschel, M., Jünger, M., & Reinelt, G. (1991). Optimal control of plotting and drilling machines: a case study. *Mathematical Methods of Operations Research, 35*(1), 61–84.
Grubbs, F. E. (1950). Sample criteria for testing outlying observations. *Annals of Mathematical Statistics, 21*(1), 27–58.
Hernández-Orallo, J., & Dowe, D. L. (2010). Measuring universal intelligence: Towards an anytime intelligence test. *Artificial Intelligence, 174*(18), 1508–1539.
Hernández-Orallo, J., Dowe, D. L., & Hernández-Lloreda, M. V. (2014). Universal psychometrics: Measuring cognitive abilities in the machine kingdom. *Cognitive Systems Research, 27*, 50–74.
Hibbard, B. (2011). Measuring agent intelligence via hierarchies of environments. In J. Schmidhuber, K. R. Thórisson, & M. Looks (Eds.). *Artificial general intelligence* (Vol. 6830, pp. 303–308). Lecture Notes in Computer Science.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Univ. of Michigan Press.
Iantovics, L. B., & Zamfirescu, C. B. (2013). ERMS: an evolutionary reorganizing multiagent system. *Innovative Computing, Information and Control, 9*(3), 1171–1188.
Jaradat, G. M., & Ayob, M. (2010). An elitist-ant system for solving the post-enrolment course timetabling problem. *Communications in Computer and Information Science, 118*, 167–176.
Johnson, B. R., Borowiec, M. L., Chiu, J. C., Lee, E. K., Atallah, J., & Ward, P. S. (2013). Phylogenomics resolves evolutionary relationships among ants, bees, and wasps. *Current Biology, 23*(20), 2058–2062.
Kattan, A., & Ong, Y. S. (2015). Surrogate genetic programming: A semantic aware evolutionary search. *Information Sciences, 296*, 345–359.
Kelley, K., & Preacher, K. J. (2012). On effect size. *Psychological Methods, 17*(2), 137–152.
Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection*. MIT Press.
Langley, P., Laird, J. E., & Rogers, S. (2009). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research, 10*(2), 141–160.
Leeuwen, Jan van (1998). Algorithms and complexity. In Jan van Leeuwen (Ed.). *Handbook of theoretical computer science* (Vol. A). Elsevier.
Lenstra, J. K., & Kan, A. H. G. R. (1974). *Some simple applications of the traveling salesman problem* BW 38/74. Amsterdam: Stichting Mathematisch Centrum.
Liang, W., Hu, Y., & Kasabov, N. K. (2015). Evolving personalized modeling system for integrated feature, neighborhood and parameter optimization utilizing gravitational search algorithm. *Evolving Systems, 6*(1), 1–14.
Lilliefors, H. (1967). On the Kolmogorov-Smirnov test for normality with mean and variance unknown. *Journal of the American Statistical Association, 62*, 399–402.
Lilliefors, H. (1969). On the Kolmogorov-Smirnov test for the exponential distribution with mean unknown. *Journal of the American Statistical Association, 64*, 387–389.
Lughofer, E., Cernuda, C., Kindermann, S., & Pratama, M. (2015). Generalized smart evolving fuzzy systems. *Evolving Systems, 6*(4), 269–292.
Mamei, M., Zambonelli, F., & Leonardi, L. (2003). Co-fields: towards a unifying approach to the engineering of swarm intelligent systems. *LNCS, 2577*, 68–81.
Mann, P. S. (1995). *Introductory statistics* (2nd ed.). Wiley.
Martinovic, G., & Bajer, D. (2012). Elitist ant system with 2-opt local search for the traveling salesman problem. *Advances in Electrical and Computer Engineering, 12*(1), 25–32.
Marusteri, M., & Bacarea, V. (2010). Comparing groups for statistical differences: how to choose the right statistical test? *Biochemia Medica, 20*(1), 15–32.
Nick, T. G. (2007). Descriptive Statistics. In W. T. Ambrosius (Ed.), *Methods in Molecular Biology 404. Topics in Biostatistics* (pp. 33–52). Totowa, NJ: Humana Press Inc..
Plante, R. D., Lowe, T. J., & Chandrasekaran, R. (1987). The product matrix traveling salesman problem: An application and solution heuristics. *Operations Research, 35*, 772–783.
Rahat, A. A. M., Everson, R. M., & Fieldsend, J. E. (2015). Hybrid evolutionary approaches to maximum lifetime routing and energy efficiency in sensor mesh networks. *Evolutionary Computation Fall, 23*(3), 481–507.
Razali, N., & Wah, Y. B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. *Journal of Statistical Modeling and Analytics, 2*(1), 21–33.
Ross, S. M. (2003). Peirce's criterion for the elimination of suspect experimental data. *Journal of Engineering Technology, 20*(2), 1–12.
Rotar, C. (2014). The endocrine control evolutionary algorithm: an extensible technique for optimization. *Natural Computing, 13*(1), 97–117.
Russell, S. J., & Novig, P. (1995). *Artificial intelligence: A modern approach*. Prentice Hall.
Schreiner, K. (2000). Measuring IS: toward a US standard. *IEEE Intelligent Systems and their Applications, 15*(5), 19–21.
Shapiro, S. S., & Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika, 52*(3–4), 591–611.
Stephens, M. A. (1974). EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association, 69*, 730–737.
Stephens, M. A. (1986). Tests based on EDF statistics. In R. B. D'Agostino & M. A. Stephens (Eds.), *Goodness-of-fit techniques*. New York: Marcel Dekker.
Stigler, S. M. (1978). Mathematical statistics in the early states. *The Annals of Statistics, 6*(2), 246.
Stoean, C., & Stoean, R. (2009). Evolution of cooperating classification rules with an archiving strategy to underpin collaboration. In H. N. Teodorescu, J. Watada, & L. C. Jain (Eds.), *Intelligent systems and technologies, series studies in computational intelligence* (pp. 47–65). Springer, 217.
Stoean, C., Preuss, M., Stoean, R., & Dumitrescu, D. (2010). Multimodal optimization by means of a topological species conservation algorithm. *IEEE Transactions on Evolutionary Computation, 14*(6), 842–864.
Stoean, R., Preuss, M., Stoean, C., El-Darzi, E., & Dumitrescu, D. (2009). Support vector machine learning with an evolutionary engine. *The Journal of the Operational Research Society, Data Mining and Operational Research: Techniques and Applications, 60*(8), 1116–1122.

Stradner, J., Thenius, R., Zahadat, P., Hamann, H., Crailsheim, K., & Schmickl, T. (2013). Algorithmic requirements for swarm intelligence in differently coupled collective systems. *Chaos, Solitons & Fractals, 50* (100), 100–114.

Theiner, G., Allen, C., & Goldstone, R. L. (2010). Recognizing group cognition, special issue on extended mind. *Cognitive Systems Research, 11*(4), 378–395.

Winklerová, Z. (2013). Maturity of the particle swarm as a metric for measuring the collective intelligence of the swarm. *Advances in Swarm Intelligence, Lecture Notes in Computer Science, 7928*, 40–54.

"World TSP Tour". http://www.math.uwaterloo.ca/tsp/world Accesses on 10.05.2016.

Yao, X. (1999). Evolutionary computation comes of age. *Cognitive Systems Research, 1*(1), 59–64.

Zerbet, A., & Nikulin, M. (2003). A new statistics for detecting outliers in exponential case. *Communications in Statistics: Theory and Methods, 32*(3), 573–584.

# On Investigating the Cognitive Complexity of Designing the Group Decision Process

**Constantin-Bala Zamfirescu[1], Luminita Duta[2], Barna Iantovics[3]**

[1] Lucian Blaga University of Sibiu, Faculty of Engineering,
Department of Computer Science and Automatic Control,
Emil Cioran 17, Sibiu, Romania,
zbc@acm.org

[2] University Valahia of Targoviste, Faculty of Electrical Engineering,
Department of Computer Science and Automation,
Unirii Ave., 18-20, Târgovişte, Romania,
duta@valahia.ro

[3] Petru Maior University of Targu Mures, Faculty of Sciences and Letters,
Department of Mathematics and Informatics,
Nicolae Iorga 1, Targu Mures, Romania,
ibarna@science.upm.ro

**Abstract:** The paper investigates the cognitive complexity associated with the design of **group decision processes** (GDP) in relation with some basic contextual factors such us task complexity, users' creativity and problem space complexity. The analysis is done by conducting a socio-simulation experiment for an envisioned software tool that acts as collaborative environment for the GDP design. The simulation results provide some insights on how to engineer context-adaptable functionalities aiming at minimizing the cognitive complexity associated with the GDP design. Although the research is carried out for a specific case, namely the GDSS technology, the results may be easily replicated for any sort of collaborative working environment where the cognitive complexity associated with its effective use is playing an important role.

**Keywords:** group decision support systems, facilitation, social simulation.

## 1. Introduction

**Group Decision Support System** (GDSS) is an interactive computer-based environment that supports concerted and coordinated team effort towards completion of joint tasks [1]. This collaborative environment is made up from a collection of highly configurable tools (i.e., brainstorming, voting and ranking, multi-criteria analysis, action planning, agenda setting etc.) which require a high level of expertise for an effective use to support complex decisions [2]. The strong relationship between the GDP outcome and the presence of a skilful facilitator to properly configure the available tools is thoroughly presented in many field studies of GDSS research [3]. Nevertheless, the presence of a scarce resource, such as a skilful facilitator, rapidly becomes the most demanding challenge in the wide spread of GDSS technology in organizations. To reduce the dependence on the facilitator, the participant-driven GDSS was proposed as a promising direction to leverage the skills and abilities of each group member [4]. However, this approach is highly constrained by the cognitive complexity associated with the construction, coordination and execution of GDP by inexperienced users.

To overcome the problem of cognitive complexity Briggs and de Vreede [5] have introduced the thinkLet (TL) concept, defined as a discrete decomposition unit that integrates a specific software tool, its configuration and a script specifying the proper usage of the tool. Consequently a TL may be considered a predefined interaction protocol among the GDSS's users, an interaction mediated and structured by a tool from the GDSS software package. As a result, the GDP is structured as a flow of discrete interactions, each of them being reflected in the specific TL that codifies the essential knowledge to execute collaborative processes.

The paper investigates the complexity associated with the GDP design in relation with some basic contextual factors such us the problem complexity, the users' creativity and the problem space complexity. The remaining part of the paper is organized as it follows. The next section describes the main components of an envisioned collaborative software tool that act as a collaborative working environment for the GDP design. These components are implemented and

tested in a socio-simulation experiment described in Section 3. As in many field studies of GDSS research, the experimental results show clear self-organizing capabilities as regards the longitudinal use of the collaborative environment to design the GDP, but simultaneously a high dependability of performance on the contextual factor. From the engineering standpoint of constructing purposeful facilitation tools for e-meetings, these results are discussed and concluded in the last section.

## 2. The Simulation Model the GDP design

The socio-simulation model is developed from the perspective of envisioning a collaborative software tool that is used by the GDSS's users to co-design the GDP. From this standpoint, the tool acts as a stigmergic environment that integrates and coordinates, on a social-scale, the facilitation knowledge of the GDSS's users. Conceptually this perspective is similar with the way in which, for instance, a collaborative CAD software is used to coordinates and integrates the specific engineering knowledge in the field architectural design [6]. Consequently, the socio-simulation basically mimics the users' conceptual 'navigation' over the semantic structure of the problem space composed of TLs. It implies the design of a population of agents and a shared environment where the agents are localized and moved over it. For the collaborative modelling of GDP the stigmergic environment is the software tool which support the manipulation of the conceptual problem space that comprise all the TLs discovered and documented by the users community (so far there are over 70 TLs acknowledged in literature [7]), while the agents are the users responsible to define, execute and evaluate the GDP (a path through the conceptual space of the available TLs).

### The semantic environment for the GDP design

According to Parunak [8], a stigmergic environment assumes the definition of three main components: 1) the topology, 2) the states, and 3) the process. Structurally, the *topology* is usually represented as a fully connected weighted graph that codifies the facilitation knowledge of group decision (Figure 1). This knowledge presumes correlated information among the users and the TLs, reflecting the users' evaluation of the TL's performance (a node in the graph) relative to a problem type. The performances are stored for each "problem type" in a variable associated with each edge of the graph. The "problem type" is simply codified through a unique ID to distinguish among different performances when they are read, during the modelling phase of the GDP, or modified, after the GDP has been executed and evaluated by the agents. Evaluation of a GDP model entails a subjective assessment of the model, after its execution, against some performance criteria that quantifies the efficiency, effectiveness and satisfaction with the GDP. Note that for the rationale of our simulation the composite criteria to quantify the TL's performance are irrelevant, it just serves as a unified denominator to measure the GDP outcome.

The performances from all the graph's edges describe the state of the environment over time. Usually, the environment executes a set of processes on the variables (as aggregation end evaporation in the case of ants [8]). For the purpose of our simulation, we apply a simple additive rule to simulate the aggregation of *performances*. After the evaluation of a GDP model that corresponds to a certain problem type, a path through a number of $n$ nodes $TL_1,\ldots,TL_n$, the aggregation rule may takes the following form:

$$P_{j,k}(TL_k,t) = P_{j,k}(TL_k,t\text{-}1) + P_{j,k}(TL_k)/\lambda \qquad (1)$$

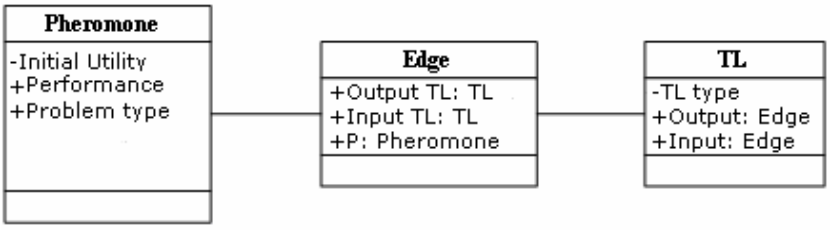



**Figure 1.** The simplified topology for the conceptual environment represented in UML.

where *t* represents the temporal component of the model which is incremented by one for each successive use of the GDSS - it practically corresponds to a simulation tick; *k* is the TL's identification index from the set of TLs used to model the GDP; $P_{j,k}(TL_k)$ – is the *performance* of the *k*-th TL evaluated from the side of TL *j*; $P_{j,k}(TL_k,t)$ and $P_{j,k}(TL_k,t\text{-}1)$ are the new and previous values of the *performances* stored on the edge between the TLs *j* and *k*; and $\lambda$ is a tuning parameter, arbitrary chosen, to weight the impact of the last evaluation.

### The agents' behaviour over the semantic environment

The *agents* are users who interact with the collaborative tool to design a GDP. Conceptually, at any time an agent is "located" in a node (TL) of the conceptual problem space, performing one of the following basic actions:

- evaluating the preference for the next possible TL (or TLs) that are going to be executed given the current context of the GDP implementation;
- selecting the next best TL (or a group of TLs) for further completing the GDP model;
- executing the TL (or the group of TLs) from the model, and finally;
- assessing the *performance* for the executed TLs.

The assessment activity is simulated using the formula (1), while the first three actions with Luce's selection axiom [9]:

$$p_{jk} = e^{P_{jk}(TL_k)/T} / \sum_{i=1}^{m} e^{P_{ji}(TL_i)/T} \qquad (2)$$

where $p_{jk}$ represents the *preference* for an alternative TL, i.e. the selection probability of the TL *k* from the TL *j*; *i* is the index of TLs connected from the side of node *j* (in fact all the *m* TLs available in the problem space as long the graph is fully connected); and *T* is a positive subunitary parameter used to define the deviation from a pure rational behaviour (for T = 1 we have a random selection behaviour, while for T = 0 a deterministic one).

The above formula is the most common model of stochastic decisions due to its correlation with the psycho-social observations of human behaviour in several domains. As a result of normalization, the preferences for the unexploited TLs are diminishing after each *performance* update. This mechanism replicates the pheromone evaporation process of the real ants (e.g. even if a TL has been positively evaluated after an execution of a GDP model, the associated preference will decrease once a better alternative is discovered and more frequently used). Under complex circumstances when TL's selection depends on other users, or the *performances* available on the edges are uncertain or incomplete, or there is impossible to evaluate the *performance* of a TL due to any real constraint (i.e. temporal, cognitive, economic, etc.), we consider the user who models the GDP to have limited cognitive capacity (bounded rationality). Note that Luce's selection axiom does not specify the reasons for the "bounded rationality"; instead, it tries to generalize the selection behaviour of human decision-makers through the parameter T which may be interpreted as the evaluation costs or uncertainty associated with the quantification of TLs' performance (on one side, when T=0, there is no uncertainty associated with the TL selection, while on the other side, when T=1, there is a completely random selection).

## 3. Experimental Results

To evaluate the cognitive complexity for modelling the GDP we conducted a virtual experiment implemented in the NetLogo multi-agent simulation environment [10] (Figure 2). Note that the NetLogo implementation includes some additional variables which are beyond the scope of this paper.

The experiment presumes that users are facilitating the e-meeting by trying to co-define, from a metacognitive stance, the GDP model for a problem type. Defining the GDP model implies the conceptual navigation in the problem space in order to find the best sequence of TLs that maximise the model's performance. The model's performance is simply computed by averaging the "Initial Utility" values for the TLs that are composing the GDP. These values are randomly chosen and assigned when the simulation is initialized.

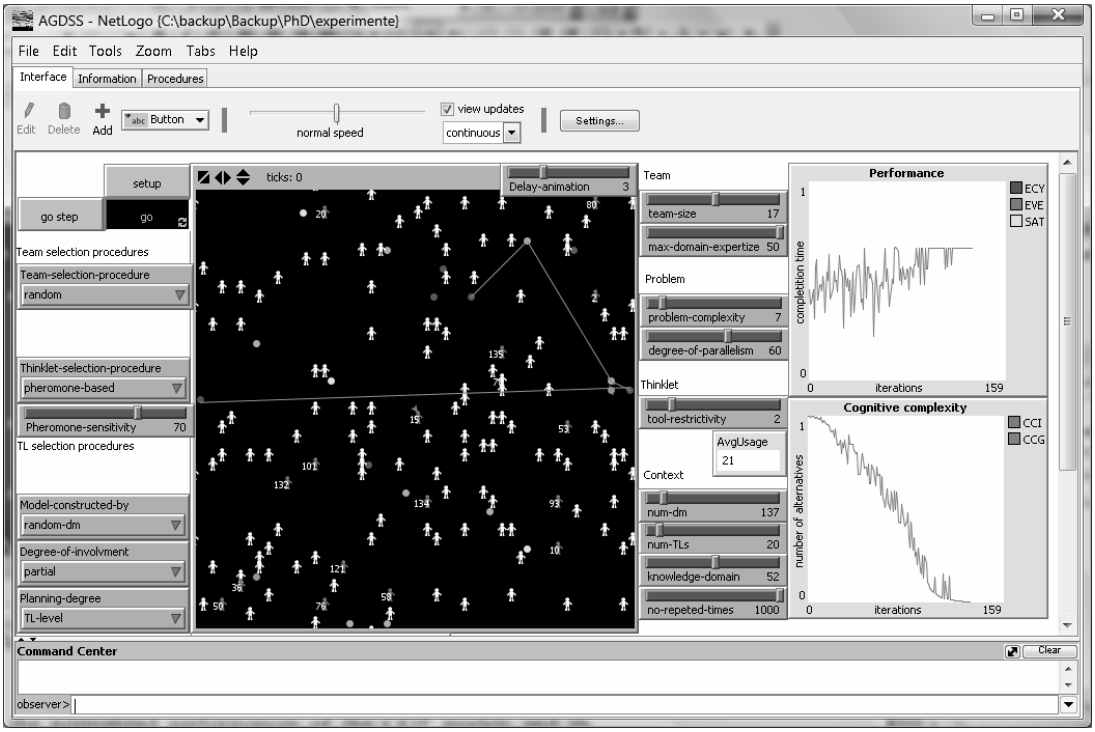



**Figure 2.** The interface of the NetLogo simulation environment for the implemented experiment.

The section presents the normalized entropy for 100 successive explorations (iterations) in relation with three factors that potentially could impact over the GDP models' performance:

- *problem complexity* (PC) – defined as the number of distinct TLs that compose a GDP model;
- *social temperature* (T) – which stands for the T parameter as defined in Eq(2); and
- *complexity of the problem space* (PS) – defined as the total number of TLs that compose the GDP modelling problem space.

An exploration stands for a complete execution cycle of a GDP. It includes three consecutive phases: 1) finding a suitable model through the successive selection (using the Eq(2)) of TLs that compose the GDP model for a given problem type; 2) executing the identified model and assessing its performance by reading and averaging the predefined performance values of all the TLs that compose the GDP model; 3) assessing the model by updating the performances values (using the Eq(1)).

The statistics are aggregated from 30 experiments for a relatively simple set of experimental values for the observed parameters.

The auto-organization of relations between TLs (i.e. the performance update after successive evaluations) entails a decrease of freedom due to the emergence of contextual constraints that reduce, in time, the probability to select some TLs (i.e. the preference for the available TL as defined in Eq(2)). For a problem type, the degree of freedom corresponds to the probabilistic distribution of preferences for the available alternatives that is equivalent with the Shannon normalized entropy [11][12]. The Shannon normalized entropy for the selection of a TL is given by:

$$E(p_{j,k}) = -\sum_{k=1}^{m} p_{j,k} \ln(p_{j,k}) / \ln(m) \qquad (3)$$

where $p_{jk}$ - represents the preference, the selection probability of the TL $k$ from the TL $j$; $i$ – is the index for the TLs connected from the node $j$ (in fact, all the $m$ TLs available in the problem space).

When the recorded performances are equal for all the available modelling alternatives, the user is considering the entire problem space when he selects a feasible TL (the probabilities from the Eq(2) being equally distributed entail an entropy equal with 1). Contrary, when the recorded *performances*

favour a single alternative, the user will have no freedom in the selection of the best TL (all the probabilities from formula (2) being 0 excepting the best alternative that is 1, entails an entropy equal with 0). Thus, the entropy associated with TL's selection is a measure of cognitive complexity for modelling the GDP. Moreover, it is a local metrics that can be computed for each TL's selection activity for modelling the GDP.

### The impact of *problem complexity* experimental variable over the cognitive complexity of GDP design

Figure 3 shows the cognitive complexity associated with the GDP modelling for different values of the PC experimental variable. The data is obtained for a problem space composed of 30 TLs and T=0.7. Since this measure is computed on the basis of the local data for each selection action (the performances available on the outward edges from the current TL), the figure corresponds to the average of entropies for all the TL selection actions needed to complete the GDP model (3, 5 and 7 successive TLs, depending on the value assigned to the PC experimental variable).

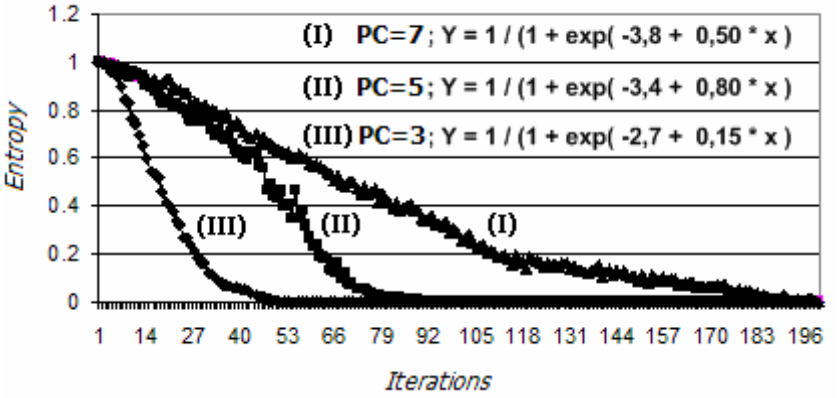



**Figure 3**. The normalized entropy of the GDP design for different problem complexities.

The data from the Figure 3 shows that problem complexity has a great impact over the entropy (around 190 iterations are required to decrease the entropy close to 0 for a PC=7, while less than 5 iterations are needed for a PC=3). These results prove that for complex problems there is an increasing need for experimentation, learning and creative use of the GDSS. At the same time, they contrast with the real use of GDSS in organizational settings where the complex problems are often less frequent and consequently there is not an adequate amount of opportunities to explore the GDSS functionalities. On the other hand, problem complexity concerns the users' satisfaction in the GDP design as well. PC is recognized by many authors to often be a subjective factor that measures the availability of relevant information [13]. The more informed/ predictable the GDP design is (i.e. the individual entropy smaller) the smaller is the subjective users' perception over the problem complexity. Consequently, the cognitive complexity for complex problems may be lessening by incorporating functionalities that provide relevant information that minimize the problem space for each discrete activity of GDP modelling. As for any design tool, these basic activities are the *selection* and *elaboration* of a model.

### The impact of *social temperature* experimental variable over the cognitive complexity of GDP design

Figure 4 shows the cognitive complexity associated with the GDP design for different values of *T*. The data is obtained for a problem space and a PC composed of 30 and 5 TLs respectively. The performances are better for higher values of T as a result of exhaustive exploration of the problem space. Consequently, when the design problem for a GDP is in the learning phase, it is preferably to encourage a creative use of the GDSS by neglecting the suggestions offered as a result of computing the collective preferences. Obviously, this issue presumes a high frequency of that problem

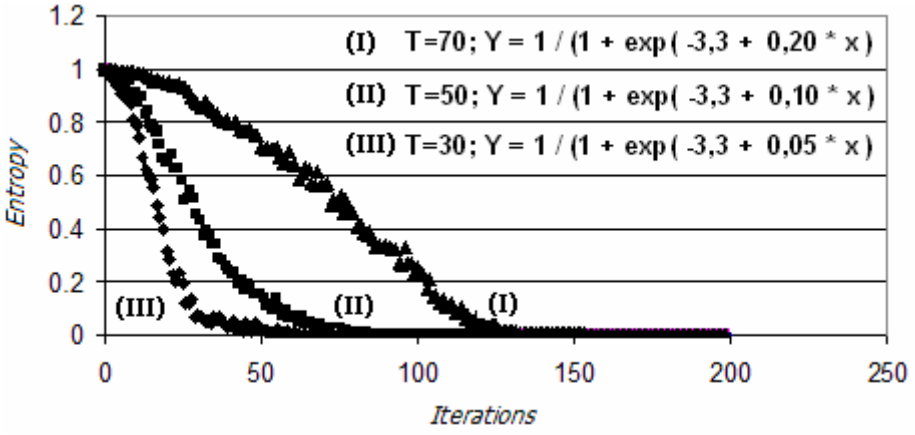



**Figure 4**. The normalized entropy of the GDP design for different T values.

type and a long-term use of the GDSS in organization to compensate the extra time required for experimentation purpose as depicted in Figure 5. Note that as long the T parameter measures the degree in which preferences are considered by the users during the GDP design, at the same time it may be used as a *post factum* measure to quantify the users' creativity.

## The impact of *problem space complexity* experimental variable over the cognitive complexity of GDP design

Figure 6 shows the cognitive complexity associated with the GDP modelling in a problem space with a different number of TLs. The data is obtained for a simple problem type composed of three TLs with T= 0.7. One can

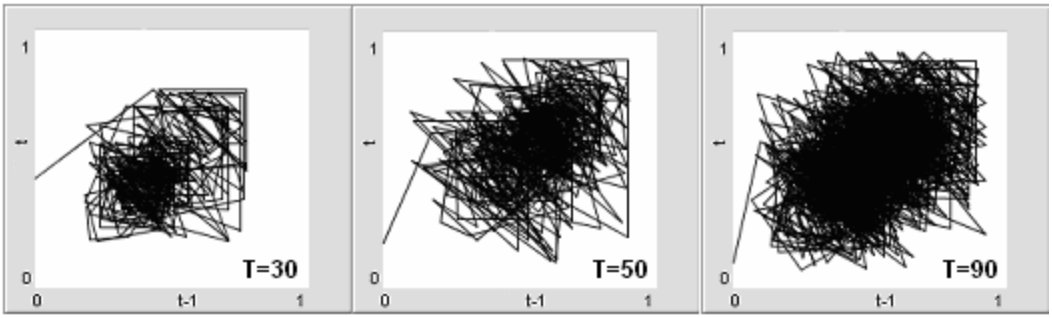



**Figure 5.** The transition maps of the GDP performance for different T values.

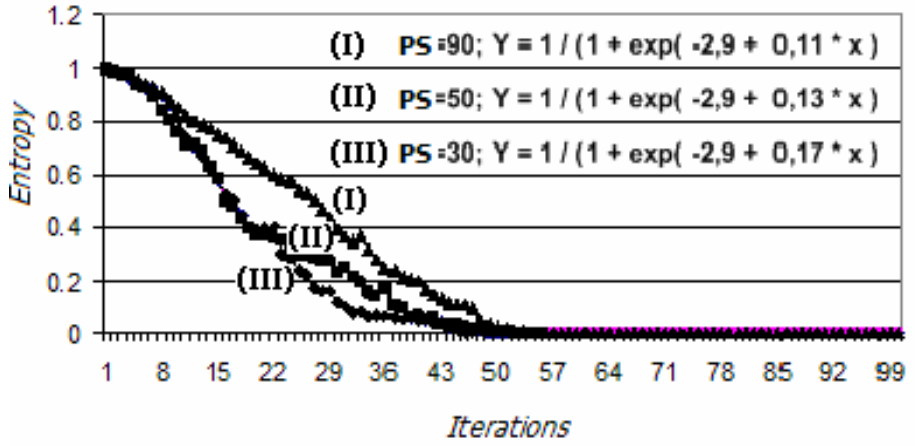



**Figure 6**. The normalized entropy for the GDP design in a problem space of different dimensions.

notice that the complexity of the problem space has basically no impact over the convergence of the entropy function. This is one of the core arguments for employing stigmergic coordination mechanisms for global optimization problems which remains scalable and effective in open, dynamic and uncertain environments. On the other hand, an increase of available TLs for modelling the GDP will automatically result in an increase of alternatives to model it. In the GDSS research field has been experimentally shown that, as the number of decision alternatives are growing, the decision makers are tempted to consider less of them [14]. This implies an accelerated discrimination of the possible alternatives through the intensification of the GDP model evaluation.

## 4. Conclusions

The work from this paper investigated some of the basic contextual factors (such us the problem complexity, the users' creativity and the problem space complexity) that have a significant impact over the cognitive complexity associated with GDP design in e-meetings. The investigation has been conducted by implementing and testing in a socio-simulation experiment an envisioned collaborative software tool that acts as a stigmergic environment for modelling the GDP.

The results extend the conclusions presented in [15][16], showing that the dominant factor for the wide adoption of GDSS technology in real organizations still remains the problem complexity. It may be lessening by incorporating functionalities that provide relevant information for the GDP design (i.e. the knowledge resulted from the subjective evaluation of each GDP from a large community of users) that entails a greater need for experimentation, learning and creative use of the GDSS. Moreover, the performances are better for higher values of the *social temperature* as a result of exhaustive exploration of the problem space. Consequently it is preferably to encourage a creative use of the GDSS when the GDP modelling problem is in the learning phase. Conversely, the complexity of the problem space has basically no impact over the cognitive complexity associated with the GDP design. This shows why the emergent functionalities of a facilitation tool for the GDP design should be engineered around some simple stigmergic coordination mechanisms.

## REFERENCES


1. DESANCTIS, G., B. GALLUPE, **A Foundation for the Study of Group Decision Support Systems**, Management Science, 1987, pp. 589-609.
2. FILIP F. G., **Decision Support and Control for Large-scale Complex Systems**, Annual Reviews in Control, vol. 32(1), 2008, pp. 61-70.
3. NIEDERMAN F., C. M. BEISE, P. M. BERANEK, **Issues and Concerns about Computer-Supported Meetings: The Facilitator's Perspective**, MIS Quarterly, vol. 20(1), 1996, pp. 1-22.
4. HELQUIST J. H., J. KRUSE, M. ADKINS, **Developing Large Scale Participant-Driven Group Support Systems: An approach to Facilitating Large Groups**, Proceedings of the First HICSS Symposium on Field and Case Studies of Collaboration, IEEE Computer Society Press, Los Alamitos, CA, 2006.
5. BRIGGS R. O., G. J. DE VREEDE, J. F. NUNAMAKER JR., **Collaboration Engineering with ThinkLets to Pursue Sustained Success with Group Support Systems**, Journal of Management Information Systems, vol. 19(4), 2003, pp. 31-63.
6. CHRISTENSEN L. R., **The Logic of Practices of Stigmergy: Representational Artifacts in Architectural Design**, Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work, ACM, New York, NY, 2008, pp. 559-568.
7. DE VREEDE G. J., R. O. BRIGGS, G. L. KOLFSCHOTEN, **ThinkLets: A Pattern Language for Facilitated and Practitioner-Guided Collaboration Processes**, International Journal of Computer Applications in Technology, vol. 25, 2006, pp. 140-154.
8. PARUNAK. H. V. D., **A Survey of Environments and Mechanisms for Human-Human** Stigmergy, Lecture

Notes on Artificial Intelligence, vol. 3830, Springer, 2006, pp. 163-186.

9. LUCE D., **Individual Choice Behaviour**, Wesley, New York, 1959.
10. WILENSKY U., **NetLogo**, Center for Connected Learning and Computer-Based Modeling, Northwestern University, http://ccl.northwestern.edu/netlogo/, 1999.
11. GUERIN S., D. KUNKLE D., **Emergence of Constraint in Self-Organizing Systems**. Nonlinear Dynamics, Psychology, and Life Sciences, vol. 8(2), 2004, pp. 131–146.
12. PARUNAK V. H. D., S. BRUECKNER, **Entropy and Self-Organization in Multi-Agent Systems.** Proceedings of the Fifth International Conference on Autonomous Agents, Montreal, Canada, 2001, pp. 124–130.
13. BYSTRÖM, K., K. JÄRVELIN, **Task Complexity Affects Information Seeking and Use**, Information Processing & Management, vol. 31, 1995, pp. 191-213.
14. POOLE M. S., J. ROTH, **Decision Development in Small Groups IV: A Typology of Group Decision Paths**, Human Communication Research, vol. 15(3), 1989, pp. 323-356.
15. C.B. ZAMFIRESCU, **An Agent-Oriented Approach for Supporting Self-Facilitation in Group Decisions**, Studies in Informatics and Control, vol. 12, 2003, pp. 137-148.
16. ZAMFIRESCU, C.B., F.G. FILIP, **Swarming Models for Facilitating Collaborative Decisions**, International Journal of Computers, Communication and Control, vol. V, 2010, http://journal.univagora.ro/download/pdf/397.pdf , pp. 125-137.

# Societal Intelligence – A New Perspective for Highly Intelligent Systems

László Barna Iantovics[1(✉)], László Szilágyi[2,3], and Camelia-M. Pintea[4]

[1] Department of Informatics, Petru Maior University of Tîrgu-Mureş, Tîrgu-Mureş, Romania
ibarna@science.upm.ro
[2] Budapest University of Technology and Economics, Budapest, Hungary
[3] Sapientia University of Transylvania, Tîrgu-Mureş, Romania
[4] Technical University of Cluj-Napoca, Cluj-Napoca, Romania

**Abstract.** A novel concept of intelligence called "societal intelligence" and its related architecture for solving complex problems are introduced. The idea is based on what we consider on the "intelligence of human society". For illustrative purposes, a case study is realized, which involves the solution of a difficult problem in a societal multi-agent system where the agents operate in an unknown environment. In the simulated robotic mobile multi-agent system, the agents adapt their movement control in the environment based on some global knowledge constructed by the system. Besides the proposed architecture, a novelty presented in the paper is the demonstration that even in a simplified knowledge-based multi-agent system, if the principles of societal intelligence are followed, a powerful global intelligence emerges at the system's level.

**Keywords:** Hybrid knowledge base · Collective intelligence · Neural network · Learning · Cognitive multi-agent system · Complex system

## 1 Introduction

In this paper we present some novel concepts of intelligence as a property of a knowledge-based multi-agent system (MAS) - that allow the solving of complex problems. Specifically, we introduce a type of intelligence namely "societal intelligence" inspired by the "emergent intelligence of the human society". Furthermore, based on this concept, we introduce a novel MAS architecture. The overall concept of intelligence is supplemented by other related concepts encompassing different types of intelligence: "individual intelligence", "individual influenced intelligence", "global intelligence base", "emergent global intelligence" and "environment intelligence". We would like to note here that the term of societal intelligence has already been used in some studies with other meanings.

The paper is organized as follows. Section 2 presents several considerations related to the knowledge-based intelligent systems (KBIS); The newly proposed societal intelligence and its architecture are described in Sect. 3; Sect. 4 shows the realized case study, its results, and a statistical analysis. In the last section, the conclusions of the research are outlined.


S. Arik et al. (Eds.): ICONIP 2015, Part IV, LNCS 9492, pp. 606–614, 2015.
DOI: 10.1007/978-3-319-26561-2_71

## 2 Intelligent Knowledge-Based Agents

One of the main purposes of most KBIS consists in attempting to obtain improvements in problems solving while comparing it against a system that is perceived not to have any intelligence, whatsoever. Increased intelligence is usually noted by being more efficient, flexible and accurate solution of difficult problems (such as the detection of clinically relevant inconsistencies [7], for example).

We would like to note here that an important aspect that should be treated, during the development of intelligent systems is: an analysis of the necessary intelligence. Sometimes, an increased intelligence can even have disadvantages. For example, if we were to consider an extremely intelligent agent as one that uses complex specializations for processing but it must solve very simple problems.

Some researches [9,10] focused on the study of decision making in the frame of cooperative coalitions, which many times outperform the decisions of individuals that operate in isolation. In many cooperative MAS, the intelligence could be considered at the level of the whole system. The intelligence in these systems is higher than the individual member agents intelligence. There are some developed systems composed of simple agents that as a whole could be considered intelligent [4,8]. The literature has not defined effective metrics that could give a quantitative evaluation to the collective intelligence.

Kun and Galis [5] present an intelligent mobile MAS composed from simple reactive agents (with knowledge retained as a set of rules) specialized in a computer network administration. The MAS simulates the behavior of a human network administrator, however, as a whole could be considered intelligent.

A novel self-adaptive MAS called ERMS that can solve problems using genetic algorithms has been proposed in [4]. The self-adaptability of ERMS consists in the capacity to autonomously reorganize its structure based on the pattern that respects the problems transmitted for solving. The reorganizations of the system are described in a rule base constructed using an evolutionary learning algorithm. The results prove that some MAS successfully can use evolutionary algorithms in order to discover emergent patterns of reorganization for efficient solving of the undertaken problems. In case of a complex network of agents, this reorganizing behavior could be associated with the intelligence.

## 3 A New Perspective for Highly Intelligent Multi-agent Systems

We consider the complexity handling of current scientific realizations of the human society and attaining the current stage of evolution, is the result of the so called societal intelligence of the human society.

### 3.1 Characterization of the Intelligence of Human Society

Each human has its own intelligence which allows problems solving, like: specialty problems (e.g. medical diagnosis problems) using specialty knowledge

(e.g. medical diagnosis knowledge) and solution of everyday life problems using commonsense knowledge (e.g. how to keep a house clean).

A human is able to take decisions based on the: commonsense knowledge (impossible to attain by computing systems [1]); specialty knowledge, attainable by expert systems [2] and human intuition defined by the psychologist and psychiatrist C. Jung. The intuition is very specific to humans and is almost totally missing from even the systems perceived to be intelligent [3]. It allows decisions taking in difficult situations, in the context of missing or erroneous information.

The following main concepts are defined for the human society and extended to MAS in the next section.

**Definition 1.** The *societal intelligence* is the intelligence of society where there is a requirement of solving a particular set of complex problems.

**Definition 2.** The *individual intelligence* defines the capacity of an individual to solve different complex problems.

We consider the existence of distributed knowledge as the global intelligence base of a society.

**Definition 3.** The *global intelligence* base of the society encompasses the entire knowledge of a society used for problem-solving.

**Definition 4.** The *influenced intelligence* of individuals defines how they learn from the global intelligence base and to learn from other individuals.

The influenced intelligence has an increasing effect to the individual intelligence. A human have the possibility to contribute to the global intelligence of the society by adding new knowledge and modifying existent knowledge. Based on these reasons the knowledge of the society is changing continuously. However, the intelligence of a human could influence the intelligence of other humans.

**Definition 5.** The *environment intelligence* is the potential of the environment to give information to individuals in order to solve problems in a more intelligent way (accurate, efficient and/or flexible).

**Definition 6.** The *emergent global intelligence* of the society is the intelligence that emerges at the human society level.

The recent results in the scientific evolution of the society are usually obtained by the collective effort of many individuals. They are based on collective intelligence, expertise, effort of many humans using distributed resources for creation/development of very complex artifacts as space rockets, airplanes etc.

### 3.2 A Novel Societal Multi-agent System Architecture

In this section we introduce the general architecture of a societal MAS, denoted with $IMG = \{IAg_1, IAg_2, \ldots, IAg_n\}$ (see Fig. 1). The agents have the same goal to solve more efficiently the problems at the level of the whole system. They belong to the same society where each agent must undertake at least one

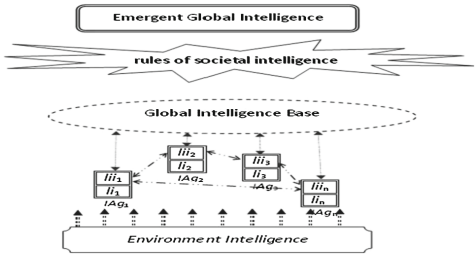



**Fig. 1.** The proposed societal multi-agent system architecture

role. The roles of an agent define the contribution of the agent to the problems solving in the frame of the MAS. To be able to undertake a role, an agent must have the necessary capacity (processing and memory) and capability (problem-solving specializations). If an agent undertakes a role it must fulfill all the commitments associated with that role (corresponding to the role it must be able to fulfill different functionalities in the frame of society). We denote with $Roles = \{rl_1, rl_2, \ldots, rl_m\}$ the existent roles in $IMG$. The notions society, role, commitment are already defined in [4].

Each agent have an "individual intelligence" (Definition 2) that defines its intelligence in problems-solving and contribution to the fulfilling of different functionalities of the MAS.

We denote with $li_k$ the individual intelligence of the $IAg_k$ agent, $li_k = locint(IAg_k)$. $li = [li_1, li_2, \ldots, li_n]$ denotes the individual intelligence of all agents from $IMG$.

The "global intelligence base" (Definition 3) consists in a specific collective intelligence formed at the level of the MAS where the agents belong. It contains some global data, information, knowledge and metaknowledge that govern the $IMG$ system coherent, efficient and flexible operation. The existent knowledge in the global intelligence base may have different representations, like: rule base, neural network, semantic net etc. or combinations of these. The content of the global intelligence base is available to the agents, they must be able to read and sometimes to write data onto it. Parts of the global intelligence base are also detained by some of the agents from the system. There exists knowledge detained by agents that is not included in the global intelligence base.

Each agent has an "influenced intelligence" (Definition 4) defining how to use the environment intelligence, learning from other agents and from the global intelligence base. Using the influenced intelligence an agent may be able to increase its individual intelligence. Improvements given by the influenced intelligence may be given by learning, adaptation and/or evolution. Using its influenced intelligence an agent should be able to extend and sometimes modify the global intelligence base and also increase the individual intelligence of other agents. We denote with $liii_k$ the influenced intelligence of the $IAg_k$ agent, $liii_k = locinfint(IAg_k)$. $liii_k$ may increase the individual intelligence $li_k$ of $IAg_k$; $li_k = fincr(li_k, liii_k, inf_p)$; $fincr$ describes how the old individual intelligence and the influenced intelligence emerge

in a new individual intelligence; $inf_p$ is data/information and/or knowledge used for the new individual intelligence. The individual influenced intelligence of agents is denoted by $lii = [lii_1, lii_2, \ldots, lii_n]$.

We mention a difficult medical case, the diagnosis of a patient with comorbidity when the necessity of the individual influenced intelligence is highly motivated. The physician in order to elaborate a diagnostic needs the assistance of an agent able to take decisions using its influenced intelligence by consulting some cognitive agents. The "intelligence of the environment" (Definition 5) sometimes chooses existent methods or combines more methods. An agent may be able or not to use emergent information/data offered by the environment.

To illustrate the concept of environment intelligence is considered the existence of ordering information related to a large set of numbers. If it is known that the numbers are ordered it can be applied the binary search method. The minimal "intelligence" - called intelligence for illustrative purposes only-in the environment (the information: the numbers are ordered), allows the improvement of the problem solving if the problems solver agent is able to understand it (understand: the numbers are ordered) and use this information (apply the most appropriate detained search technique).

The emergent global intelligence (Definition 6) is the problem-solving intelligence of the $IMG$ system that emerges at the level of the whole system.

The societal intelligence in a societal MAS is defined by some specific rules that govern the organization and intelligence of the society of agents. Establishes how the global intelligence base is constructed and used by the individuals from the society. It defines the individuals personal intelligence and influenced intelligence (how they learn from each others, from the environment and from the global knowledge of the society). The application of the societal rules has as effect the emergence of a global intelligence at the society level.

## 4 Case Study of a Simple Societal Multi-agent System

Many times the intelligence of a system is considered based on capabilities like [4,6]: learning, self-adaptation and evolution. The existence of some of these capabilities does not implicitly allow a quantitative evaluation of the intelligence, they just prove its existence. We consider that the evaluation of a system's intelligence must be based on some metrics. Such metrics should be determined based on considerations that include problems solving point of view, like: difficult problems solving, capacity to treat different types of uncertainties etc.

Different studies related with robots are realized in [11]. We have implemented a simplified societal MAS. Notions related with different types of intelligence (and knowledge), like individual intelligence and individual influenced intelligence are used with the significance of intelligence just for illustrative purposes. We have simulated a two-dimensional virtual software environment where operates an $n$-element robotic mobile MAS denoted $SMG = \{RA_1 \ldots, RA_n\}$. Each of the agents has the same role to search for the same specified object. The agents using their effectors are able to move in the environment.

Using their sensors they are able to explore the environment during their movement and read-write information onto the global intelligence base. The environment map is not known to the agents. Each agent from the system uses the same method denoted $Eval$ for the objects evaluation: $Eval : Ob \rightarrow R+$. An object denoted $Ob_k$ ($Ob_k \in Ob$; $Ob$ denotes the objects' space) based on the similarities can be more or less appropriate with the value of the searched object. At each problem-solving cycle, $SMG$ must solve a single problem denoted $Pr$ of finding a specified object. $SMG$ solve in a cooperative way the problem (find the object or fail). The individual intelligence of an $RA_k$ agent consists in the followings:

- capacity to move in the environment;
- capacity to evaluate the objects in the environment;
- capacity to take into consideration the detained local history denoted $Best_k = (BestVal_k; Position_k)$; $BestVal_k$ denotes the best value found until that time; $Position_k$ denotes the position of the best value;
- Local history constructed during a problems solving.

The global intelligence base denoted $GIB$ is dynamically constructed step-by-step by all the agents during a problem's solving. $GIB$ contains the following data at each problem-solving cycle, $Global = (GlobalVal; GlobalPosition)$: $GlobalVal$ denotes the highest value found; $GlobalPosition$ denotes the global position of the best value found at system's level. The individual influenced intelligence of an agent $RA_k$ consist in the capacity to take into consideration the best solution detained at a moment of time in the global intelligence base. Each agent $RA_i$ is able to modify $GIB$ when it finds a better solution. Each agent $RA_i$ may take into consideration in a certain degree the personal best previously found during its history $PersBest_i = (BestVal_i; Position_i)$. Each agent may take into consideration to a certain degree the best location that any agent has ever found the global best $GlobalBest = (GlobalVal; GlobalPosition)$.

*Cooperative Problem Solving* presents the main steps of the $SMG$ system operation at a problem's solving cycle. A problem-solving cycle begins when a problem is submitted to be solved and finishes when is obtained a solution or the running time expires. $Gen$ denotes the admitted maximal time steps. It is not possible the theoretical determination of the necessary time, however,

```
Step 1.
@Creation of the environment and the distributed objects.
@Creation of the societal multi-agent system.
@Specification of the searched object.
Step 2.
while (Object not found) and (Gen not reached) do
 | @Cooperative search for the object in the environment
end
Step 3.
@Obtained solution is reported.
```

**Algorithm 1:** Cooperative problem solving algorithm

**Table 1.** Results of 20 simulations

| | Table 1.1. $apb = 5$, $agb = 5$ | | | | Table 1.2. $apb = 8$, $agb = 8$ | | | |
|---|---|---|---|---|---|---|---|---|
| $In$ | 20 | 40 | 60 | 80 | 20 | 40 | 60 | 80 |
| 3 | 550.8 / 8 | 503.1 / 5 | 495.0 / 4 | 490.6 / 2 | 560.9 / 9 | 510.6 / 7 | 470.5 / 5 | 495.4 / 4 |
| 2 | 510.1 / 6 | 497.7 / 4 | 430.2 / 3 | 428.2 / 3 | 532.3 / 7 | 493.1 / 6 | 469.7 / 4 | 451.5 / 4 |
| 1 | 420.0 / 3 | 390.6 / 3 | 330.8 / 2 | 326.4 / 2 | 335.0 / 3 | 312.9 / 3 | 290.6 / 2 | 260.7 / 1 |

**Table 2.** Results of the two-factor ANOVA tests, $\alpha = 0.05$

| | | Table 2.1. Problem solving time | | | Table 2.2. Number of missed solutions | | |
|---|---|---|---|---|---|---|---|
| | Fc | P-value | Fr/Fcr | Remark | P-value | Fr/Fcr | Remark |
| 1-R | 5.14 | $\approx 0.000036$ | 87.64 | $Fr > Fcr$ | $\approx 0.061$ | 4.61 | $Fr < Fcr$ |
| 2-R | 5.14 | $\approx 0.0000009$ | 303.27 | $Fr > Fcr$ | $\approx 0.00068$ | 31.2 | $Fr > Fcr$ |
| 1-C | 4.76 | $\approx 0.0023$ | 17.43 | $Fc > Fcr$ | $\approx 0.037$ | 5.54 | $Fc > Fcr$ |
| 2-C | 4.76 | $\approx 0.0018$ | 19.09 | $Fc > Fcr$ | $\approx 0.0053$ | 12.55 | $Fc > Fcr$ |

is empirically established as $Gen = 600$. In different implementations depends on factors like the environment intelligence, the initial distribution of the agents in the environment and values of different parameters of the algorithm.

The parameters values of the simulation were $|SMG| \in [20, 80]$; *environment intelligence* (with significance "degree of unknown") denoted $ei$, $ei \in \{1, 2, 3\}$; consideration of the personal best denoted $apb$, with $apb \in [0, 10]$; consideration of global best denoted $agb$, with $agb \in [0, 10]$. Based on the simplified environment we have defined three types of environment intelligence: $ei = 3$-no intelligence (randomly generated and distributed objects); $ei = 2$-very small intelligence; $ei = 1$-small intelligence.

Emergent global intelligence measured by the elaborated metric considers two criteria, the average problem solving time of more simulations (until the solution has been found) and the number of missed solutions in that simulations (the solution is not obtained in the admitted time steps $Gen$). A result of 20 problem's solving cycle of the same problem is denoted with $Result = \{Ind_1; Ind_2\}$; $Ind_1$-shows the averaged number of time steps when the solution is obtained during 20 simulations; $Ind_2$- shows how many times the solution was missing during 20 simulations. $In = ei/|SMG|$. Table 1 presents some simulation results.

In order to prove that the number of agents and the environment intelligence influences the problem solving time, verification of statistical difference we consider a two-factor analysis of variance ANOVA without replication by taking into consideration $ei$, $|SMG|$ (independent) and the problem solving time (dependent). It is also considered a two-factor analysis of variance ANOVA without replication by taking into consideration $ei$, $|SMG|$ (independent) and the number of times when the solution have been missed (dependent).

For both ANOVA analysis it is considered the significance level $\alpha = 0.05$. The results of the ANOVA tests in the Table 2 are presented (conclusions that can be obtained are based on the details presented in the remark columns). Can be concluded a decrease of problem solving time (statistical difference at $\alpha = 0.05$) based on the increase of environment intelligence and the number of agents. The misses time (statistically analyzed at $\alpha = 0.05$), is influenced by the number of agents, the environment intelligence proved influence just in some conditions.

We have observed in our research that in a highly complex societal MAS an agent could not detain by itself the maximal individual intelligence for the most optimal operation and problem solving. The agent must have influenced individual intelligence with learning capabilities, learning from other agents and from the global intelligence base. In our case-study the use of information related with the global best improves the problem solving time.

## 5 Conclusion

We have proposed a novel type of intelligence for difficult problems solving called societal intelligence, inspired by the "intelligence of the human society". For proving the effectiveness of a societal MAS, we have realized a case study. The main conclusion of the considered case study was that using the principles of societal intelligence even if the intelligence of the considered agents is very limited, in the system emerges an increased global intelligence at the system's level. Based on a comprehensive study of the scientific literature we consider that our proposal is original and will represent the basis for many future researches, including elaboration of some metrics for the measurement of complex systems' intelligence able to solve extremely difficult problems. In our approach, there are considered different types of intelligence that can be measured and based on them can be established the intelligence of a highly complex system as a whole.

## References

1. Zang, L.J., Cao, C., Cao, Y.N., Wu, Y.M., Cao, C.G.: A survey of commonsense knowledge acquisition. J. Comp. Sci. Technol. **28**, 689–719 (2013)
2. Duda, R.O., Shortliffe, E.H.: Expert systems research. Science **220**, 261–268 (1983)
3. Frantz, R.: Herbert Simon: Artificial intelligence as a framework for understanding intuition. J. Economic. Psychol. **24**, 265–277 (2003)
4. Iantovics, L.B., Zamfirescu, C.B.: ERMS: an evolutionary reorganizing multiagent system. Int. J. Innov. Comput. Inf. Control **9**, 1171–1188 (2013)
5. Kun, Y., Galis, A., Guo, X., Liu, D.: Rule-driven mobile intelligent agents for real-time configuration of IP networks. In: Palade, V., Howlett, R.J., Jain, L. (eds.) KES 2003. LNCS, vol. 2773, pp. 921–928. Springer, Heidelberg (2003)
6. Mamei, M., Menezes, R., Tolksdorf, R., Zambonelli, F.: Case studies for self-organization in computer science. J. Syst. Architect. **52**, 443–460 (2006)
7. McShane, M., Beale, S., Nirenburg, S., Jarrell, B., Fantry, G.: Inconsistency as a diagnostic tool in a society of intelligent agents. Artif. Intell. Med. **55**, 137–148 (2012)

8. Micacchi, C., Cohen, R.: A framework for simulating real-time multi-agent systems. Knowl. Inform. Syst. **17**, 135–166 (2008)
9. West, D., Dellana, S.: Diversity of ability and cognitive style for group decision processes. Inform. Sci. **179**, 542–555 (2009)
10. Zamfirescu, C.B., Duta, L., Iantovics, L.B.: On investigating the cognitive complexity of designing the group decision process. Stud. Inform. Contr. **19**, 263–270 (2010)
11. Moldovan, L.: Geometrical method for description of the 6-PGK parallel robot's workspace. In: CANS Conference, pp. 45–51. IEEE Computer Society Press (2008)

# A Novel Osmosis-Inspired Algorithm for Multiobjective Optimization

Corina Rotar[1(✉)], Laszlo Barna Iantovics[2], and Sabri Arik[3]

[1] "1 Decembrie 1918" University of Alba Iulia, Alba Iulia, Romania
crotar@uab.ro
[2] Petru Maior University, Targu Mures, Romania
[3] Istanbul Unıversity, Istanbul, Turkey

**Abstract.** Many real-life difficult problems imply more than one optimization criterion and often require multiobjective optimization techniques. Among these techniques, nature-inspired algorithms, for instance, evolutionary algorithms, mimic various natural process and systems and succeed to perform appropriately for hard optimization problems. Besides, in chemistry, osmosis is the natural process of balancing the concentration of two solutions. This process takes place at the molecular level. Osmosis's practical applications are multiple and target medicine, food safety, and engineering. However, osmosis process is not yet recognized as a rich source of inspiration for designing computational tools. At first glance, this well-known chemical process seems appropriate as a metaphor in nature-inspired computation as it can underlie the development of a search and optimization procedure. In this paper, we develop a novel algorithm called OSMIA (Osmosis inspired Algorithm) for multiobjective optimization problems. The proposed algorithm is inspired by the well-known physio-chemical osmosis process. For validation purposes, we have realized a case study in that we compared our proposed algorithm with the state-of-art algorithm NSGAII using some well known test problems. The conclusions of the case study emphasize the strengths of the proposed novel OSMIA algorithm.

**Keywords:** Nature-inspired algorithm · Multiobjective · Osmosis process

## 1 Introduction

Nature-inspired computing represents a significant research area of the Artificial Intelligence. It includes evolutionary algorithms [1], neural networks [2], cellular automata [3], emergent systems [4–6], artificial immune systems [7], membrane computing [8] and many others. Simply, nature-inspired computing is a growing field, developed by mostly imitating biological models, for the development of the computational models and techniques. Among these, a prolific research chapter includes nature-inspired algorithms for solving the multiobjective optimization problems (MOPs). Given the MOPs complexity and due to the proven potential of the nature-inspired algorithm for various complex problems, the developing and improving the nature-inspired techniques for MOPs represents a challenging task.


D. Liu et al. (Eds.): ICONIP 2017, Part IV, LNCS 10637, pp. 80–88, 2017.
https://doi.org/10.1007/978-3-319-70093-9_9

The research presented in this paper focuses on two complementary directions involved and their functionality, and secondly, the adapting of the considered natural models into powerful computing models. The palette of the natural paradigms, which underlies the development of the nature-inspired metaheuristics, is diverse and encompasses the functioning of the brain, Darwinian evolution, self-replication, collective behavior, the vertebrate immune system, cell membranes, morphogenesis, and so on. There are, moreover, a lot of other natural phenomena which could lead to the development of computational methods. Of these, we identified a physicochemical process, the osmotic process that stands out by its strength and simplicity. Nevertheless, the physicochemical processes underlying the designing of the calculation methods are reduced in number compared with the patterns inspired by the biological systems and processes. Therefore, we identify as a new research domain to the identification of the manner in which the osmosis paradigm could generate new computational models. In this paper, we present such a novel algorithm that we called OSMIA.

The upcoming part of the paper is organized as follows: in the Sect. 2 we describe the natural paradigm and propose a novel osmosis-inspired algorithm for multiobjective optimization. Section 2.3 presents the experimental results. In Sect. 3 we discuss the main results and suggest further research directions.

## 2 Design of a New Metaheuristic Inspired by Nature

### 2.1 Natural Paradigm: Osmosis Process

Osmosis is the natural process of balancing the concentration of two solutions, a process that takes place at the molecular level. The process of osmosis is described as the diffusion of a solvent (usually water) through a semi-permeable membrane from a solution with low concentration of solute (high water potential, or, Hypotonic) in a solution with higher concentration solute (low potential of water, or, Hypertonic) to a certain concentration level/gradient of the solution. It is a physical process in which a solvent moves without receiving power through a semipermeable membrane (permeable to solvent but not the solution) separating the two different solutions. This effect can be measured by increased pressure of the hypertonic solution, compared to the hypotonic solution.

For instance, if two solutions of different concentration are separated by a membrane, which is permeable only to the smaller solvent molecules but not to the larger solute molecules, then the solvent will tend to diffuse across the membrane from the less concentrated to the more concentrated solution. As in Fig. 1, having two containers that communicate through a semipermeable membrane, one with less saline water and the other one with a saline solution of a higher concentration, the molecules of the pure water will migrate into the saline solution to reduce the concentration level until it reaches a balance. This phenomenon is caused by the normal diffusion of the molecules, and not by an external force.

Osmotic pressure is the pressure that must be applied to a solution to prevent the solvent migration, in the natural sense of diffusion, through a semi-permeable membrane. Considering $\Delta h$ – the difference in height of the solution, $\rho$ – the density of the

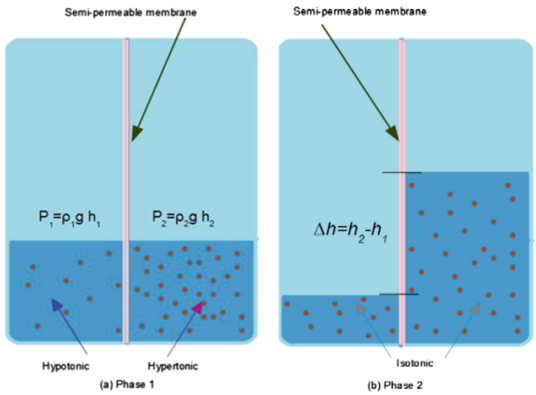



**Fig. 1.** Description of the osmosis process: before osmotic equilibrium (a); after osmotic equilibrium is attained (b)

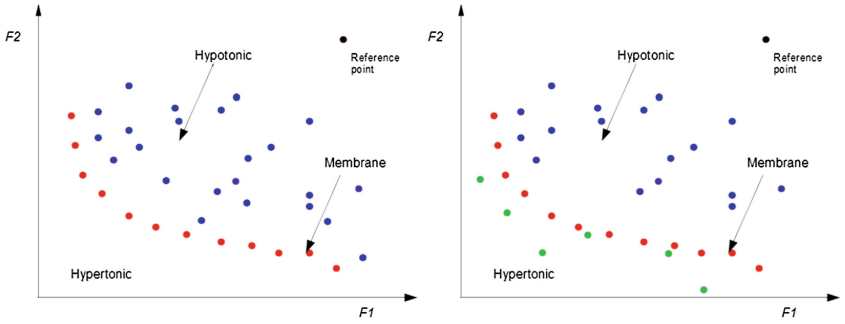



**Fig. 2.** First phase: molecules occupy the hypertonic region (*left*). During osmosis: the molecules diffuse through the membrane (*right*)

solution and $g$ – the gravitational acceleration, the osmotic equilibrium is achieved when the osmotic pressure reaches the hydrostatic pressure, as follows:

$$P_{osmotic} = \rho \cdot g \cdot \Delta h \tag{1}$$

## 2.2 OSMIA a Novel Osmosis-Inspired Algorithm

We observed that the natural process of osmosis can be considered as a model that underlies the development of a search and optimization procedure in multiobjective optimization problems. The strategy inspired by osmosis involves the management of three populations of molecules. These populations correspond to the water molecules from the hypotonic and hypertonic solutions.

Let us consider:

(a) *Hypotonic* = the set of molecules in the hypotonic environment
(b) *Membrane* = the set of molecules which form the membrane
(c) *Hypertonic* = the set of molecules in the hypertonic environment

Let us consider n – the dimension of the search space and m – the number of objectives. A molecule structure is given by the following formula.

$$mol = \left\{location_{Search}, location_{Obj}, mass, type\right\} \tag{2}$$

where: $location_{Search} = (x_1, x_2, \ldots, x_n)$ represents the location of the molecule in the search space; $location_{Obj} = (f_1, f_2, \ldots, f_m)$ represents the corresponding location in the objective space and type – represents an indicator of the actual state of the molecule (type = 0 if $mol \in Hypotonic$, type = 1 if $mol \in Membrane$, type = 2 if $mol \in Hypertonic$).

For a minimization problem, the mass of the molecule is computed as follows, signifying that those solutions, which have lower objective values, gain higher mass.

$$mass = 1 \Big/ \left(1 + \sum_{i=1}^{m} f_i\right) \tag{3}$$

In the first phase, a set of molecules is randomly generated. The size of the initial set is given *dim*. Each molecule corresponds to a possible solution in the search space. Some of these molecules form a virtual membrane. Establishing semi-permeable membrane is made after the evaluation of the solutions. The molecules are divided into two sets: Pareto non-dominated and dominated solutions. The molecules that correspond to non-dominant solutions will form the "semi-permeable membrane".

The hypotonic environment contains those molecules, which correspond to the dominated solutions. The molecules which correspond to the Pareto non-dominated solutions delimit the semi-permeable membrane. Those molecules, which diffuse through the membrane, according to the Pareto domination criterion, are classified as members of the hypertonic set. The osmotic process continues through the movement of molecules of the hypotonic environment. Hypotonic molecules diffuse toward different positions in the search space. If after the movement of a molecule, the new location of the molecule dominates the current position, this new position is retained further and the molecule is marked accordingly; otherwise, if the new position corresponds to a weaker solution, the old position is restored. A new position of a molecule is better if the corresponding candidate solution dominates the solution which corresponds to the molecule in the original location. If the new position dominates the previous one and the previous position corresponds to a non-dominated solution in current population, the molecule will be marked as a new member of the virtual membrane.

***Osmosis Procedure/Cycle***

**While** (not osmotic equilibrium) Do
  **For (**each molecule in the Hypotonic solution) Do
    New:=Move the molecule
    **if** (New molecule *is located in the hypertonic solution*) **then**
      *Hypertonic=Hypertonic* U {New}
**EndWhile**

The proposed algorithm inspired by osmosis repeats the osmotic procedure, which takes as long as the equilibrium between the hypertonic, respectively, hypotonic environments, is not reached. The osmosis is considered complete when the hydrostatic pressure balances the osmotic pressure, and therefore, the molecules will no longer flow from the hypotonic to the hypertonic fluid. Following figures describe the osmotic cycle for a bi-objective minimization problem (Fig. 3).

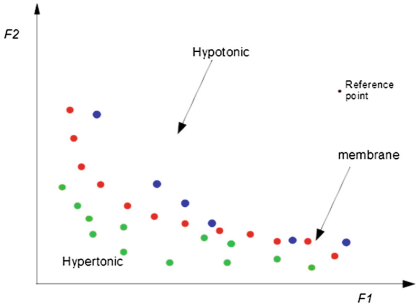



**Fig. 3.** Finale stage: osmosis process is done; the molecules occupy the hypotonic region

Let us consider: $\rho_1$, $\rho_2$, $h_1$, $h_2$ the pressures and heights of the two environments, at a certain time during the osmosis process, the equilibrium is reached when $P_1 = P_2$:

$$\rho_1 \cdot g \cdot h_1 = \rho_2 \cdot g \cdot h_2$$

Therefore, while $P_1 > P_2$, the molecules from the hypotonic environment will diffuse into the hypertonic environment.

The proposed Osmosis inspired Algorithm for Multicriteria Optimization (*OSMIA*) works as follows: while a termination condition is not true, the osmosis procedure runs and the virtual membrane is updated. The membrane consists of those molecules that correspond to the non-dominated solutions, and it is a dynamic structure, as long as the set of molecules varies along the osmosis process.

Figure 2 depicts a configuration of molecules in the objective space before the osmosis procedure starts. The reference point is computed as follows:

$$reference = (r_1, r_2, \ldots, r_m), \text{ where } r_i = \max\{f_i^j, j = 1 \ldots dim\}$$

In addition, for the hypertonic and hypotonic sets of molecules, the centroids are computed, as the arithmetic mean position of all the points in the corresponding set:

$$centroid(Hypertonic) = (H_1, H_2, \ldots, H_m)$$
$$centroid(hypotonic) = (h_1, h_2, \ldots, h_m)$$

Relative to this reference point, the pressures and heights of the two environments (hypotonic and hypertonic) can be computed. The "*heights*" of two environments are given by the Euclidian distances between the centroids of the Hypotonic and, respectively Hypertonic set and the reference point:

$$h_1 = \text{distance}(centroid(Hypotonic), reference)$$
$$h_2 = \text{distance}(centroid(Hypertonic), reference)$$

The "densities" of the two environments are computed by the following formulas:

$$\rho_1 = \frac{average\_mass_{hypotonic}}{Volume_{hypotonic}}$$
$$\rho_2 = \frac{average\_mass_{hypertonic}}{Volume_{hypertonic}}$$

The *average_mass* represents the arithmetic mean of the molecules' masses from the specific environment. The *Volume* is estimated by Ritter's algorithm [9] which finds the bounding sphere that contains all of a given molecules from the specific environment.

**Algorithm OSMIA is:**
Generate the set of molecules: **Hypotonic, Membrane, Hypertonic**
**While** (ending_condition*)
Compute: **reference**, $\rho_1$ , $\rho_2$ ,$h_1$,$h_2$ //densities and heights
**While**( $\rho_1 \cdot g \cdot h_1 > \rho_2 \cdot g \cdot h_2$ )//**Osmosis cycle**
**For** each *mol* from **Hypertonic U Hypertonic U Membrane**
*New=Move(mol)*
**If** *New dominates mol* and mol∈Membrane then
*Membrane=Membrane U {New}*
**If** *New dominates mol* and mol ∉Membrane then
*Hypertonic=Hypertonic U {New}*
**Endfor**
**EndWhile**
Update ***Membrane*** = set of non-dominated molecules
**EndWhile**
**End**
**ending_condition may refer to attaining the maximum number of cycles*

The *Move* procedure mimics the Brownian movement of the molecules within the given space. Therefore, the procedure varies the location (considered in the search space) of the current molecule.

For an intuitive approach, each molecule from the hypertonic and hypotonic sets shifts its position, with a given probability $p$, in each of the $i^{th}$ dimension of the search space. The probability of altering a specific coordinate is set to the value $p = 1/n$, where $n$ – represents the given dimensionality of the search space. The alteration of the coordinates is influenced by one randomly selected element from the membrane (non-dominated solutions).

```
FunctionMove(mol1): mol2
   Input: mol1, Output: mol2
   mol2=mol1
   Randomly select Membrane(ind)
   for each i from {1,…n}
      if rand<1/n then
         if mol2.x(i) <Membrane(ind).x(i) then
            mol2.x(i)=Membrane(ind).x(i)+ rand*(Max-Membrane(ind).x(i))
      if mol2.x(i) > Membrane(ind).x(i) then
               mol2.x(i) = Membrane(ind).x(i)-rand*(Membrane(ind).x(i)-Min)
   endfor
End
```

### 2.3 Experimental Results

In order to illustrate the performance of the proposed OSMIA algorithm, we used several well-known test problems ZDT1, ZDT2, ZDT3 [10] used in the most of the researches and a state-of-art algorithm for multiobjective optimization: NSGAII [11]. NSGAII algorithm have many significant recent applications. For example we mention the multi-production and multi-echelon closed-loop pharmaceutical supply chain considering quality concepts [12], finding patterns in protein sequences [13]. For performance assessment, we compute the hypervolume metric (HV) [14]. The hyper-volume metric corresponds to the size of the objective space, which contains the solutions that are Pareto-dominated by at least one of the members of the set. The higher the hyper-volume value is, the better outcomes the algorithm provides. Among the performance metrics, the hypervolume is popular as it captures both the convergence to the true Pareto front and the distribution over the objective space.

For an objective comparison with the popular algorithm NSGAII [11], we set the following parameters for the OSMIA: the size of molecule library is set to 50, and the maximum number of fitness evaluation is set to 10000. The most appropriate parameter values we have established experimentally. NSGAII's parameters (that are considered in different studies) are 100 individuals and 100 iterations per run. These settings assure the same maximum number of function evaluations for both algorithms. The algorithms run for 10 times and the hypervolume values are computed. The results are described in Table 1.

**Table 1.** Hypervolume: OSMIA versus NSGA2.

| Test problem | HV | Mean value | Maximum | Standard dev |
|---|---|---|---|---|
| ZDT1 | *OSMIA* | **0.847666** | 0.85257 | 3.99E−03 |
| | *NSGA2* | 0.775539 | 0.804457 | 2.01E−02 |
| ZDT2 | *OSMIA* | **0.768684** | 0.783413 | 2.31E−02 |
| | *NSGA2* | 0.579927 | 0.601901 | 2.41E−02 |
| ZDT3 | *OSMIA* | **0.634674** | 0.650786 | 2.85E−02 |
| | *NSGA2* | 0.514411 | 0.631122 | 7.66E−02 |

The results presented in Table 1, show that for each test problem we considered, the proposed algorithm performs better than NSGAII.

## 3 Conclusions

In this paper, we explored a novel metaphor in Natural Computing, i.e. the natural process of osmosis, and we proposed a new metaheuristic for multiobjective optimization. The osmosis-inspired algorithm, OSMIA, is a population-based algorithm for optimization, which mimics the process of molecules' diffusion through a semi-permeable membrane. The convergence toward the problem's solutions is guided by the virtual membrane which collects, at each cycle, the set of Pareto non-dominated solutions. The molecules from the hypotonic environment diffuse through the virtual membrane, into the hypertonic environment by using a procedure that alters the original location and simulates Brownian movement. Each diffusion cycle is considered done when the osmotic equilibrium is attained. In natural paradigm, the osmotic equilibrium has attained the concentration is the same on both sides of a semi-permeable membrane. In artificial model, the osmotic equilibrium is considered achieved when the hydrostatic pressure balances the osmotic pressure.

OSMIA was compared with state of art algorithm for multiobjective optimization, NSGAII and the results showed that our proposal performs better in all test scenarios. As further research, we propose to investigate the recognized natural metaphor for different problems and to investigate OSMIA's performance for more difficult optimization problems.

An advantage of the proposed algorithm is given by the minimum number of user-defined parameters. Excluding the number of objectives, the number of variables, and the size of the initial set of molecules (*dim*), no other parameter is needed. Therefore, the Osmosis-inspired Algorithm can be considered a parameter free technique that may solve numerous optimization problems, which involve multiple criteria. Among these, we will investigate the problem of determination of the types of degradation that may affect heritage buildings due to multiple factors. The factors that may affect the heritage buildings include physical, chemical and biological actions. Also, we will consider OSMIA algorithm for a real-world problem such as identifying the optimal strategy to manage the waste, which results from interventions on buildings.

**Acknowledgements.** The authors gratefully acknowledge the financial support provided by the Romanian National Authority for Scientific Research, CNCS – UEFISCDI, under the Bridge Grant PN-III-P2-2.1-BG-2016-0302.

## References

1. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing, vol. 53. Springer, Heidelberg (2003)
2. Rojas, R.: Neural Networks: a Systematic Introduction. Springer-Verlag New York, Inc., New York (1996)
3. Wolfram, S.: Universality and complexity in cellular automata. Phys. D: Nonlinear Phenom. **10**(1), 1–35 (1984)
4. Karaboga, D., Celal, O.: A novel clustering approach: artificial bee colony (ABC) algorithm. Appl. Soft Comput. **11**(1), 652–657 (2011)
5. Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.): ANTS 2008. LNCS, vol. 5217. Springer, Heidelberg (2008). doi:10.1007/978-3-540-87527-7
6. Karaboga, D.: An idea based on honey bee swarm for numerical optimization, vol. 200, Technical report-tr06, Computer Engineering Department, Erciyes University (2005)
7. De Castro, L.N., Timmis, J.: Artificial immune systems: a new computational intelligence approach. Springer Science & Business Media (2002)
8. Păun, G.: Computing with membranes. J. Comput. Syst. Sci. **61**(1), 108–143 (2000)
9. Ritter, J.: An efficient bounding sphere. In: Glassner, A. (ed.) Graphics Gems. Academic Press, Boston, MA (1990)
10. Zitzler, E., Deb, K., Thiele, L.: Comparison of multiobjective evolutionary algorithms: empirical results. Evol. Comput. **8**(2), 173–195 (2000)
11. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
12. Moslemi, S., Zavvar Sabegh, M.H., Mirzazadeh, A., Ozturkoglu, Y., Maass, E.: Int J Syst Assur Eng Manage (2017). doi:10.1007/s13198-017-0650-4
13. González-Álvarez, D.L., Vega-Rodríguez, M.A., Rubio-Largo, Á.: A hybrid MPI/OpenMP parallel implementation of NSGA-II for finding patterns in protein sequences. Supercomput. **73**(6), 2285–2312 (2017). doi:10.1007/s11227-016-1916-3
14. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)

International Journal of Innovative
Computing, Information and Control
Volume **9**, Number **3**, March **2013**

ICIC International ©2013 ISSN 1349-4198
pp. 1171–1188

# ERMS: AN EVOLUTIONARY REORGANIZING MULTIAGENT SYSTEM

Barna Laszlo Iantovics[1] and Constantin-Bala Zamfirescu[2]

[1]Informatics and Mathematics Department
Petru Maior University
No. 1, Iuliu Maniu St., Tg. Mures 540088, Romania
ibarna@upm.ro

[2]Lucian Blaga University
No. 10, Victoriei St., Sibiu 550024, Romania
zbc@acm.org

Received December 2011; revised April 2012

Abstract. *Development of computational systems that intelligently can solve problems represents an important research direction. Many results described in the literature prove, that the intelligence of a computational system can offer advantages in the problems solving versus a system that does not have intelligence. The adaptation is considered to be an important property of many intelligent systems. Sometimes the adaptation is realized by learning. In this paper, we propose a novel adaptive multiagent system called ERMS (Extended Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity), which uses an evolutionary learning technique in order to improve the efficiency of the undertaken problems-solving. ERMS represents an extension of a previously developed multiagent system called CCER (Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity). The adaptivity of the ERMS multiagent system, consists in the capacity to reorganize its structure based on the information available about the received problems for solving. The obtained results prove that a multiagent system successfully can use evolutionary algorithms to discover emergent patterns of reorganization for the efficient solving of the undertaken problems. In case of complex systems composed from a large number of interacting components, such emergent behavior of the systems, if have as results improvements (i.e., autonomy, efficiency and flexibility) in the problems solving it could be associated with intelligence.*

**Keywords:** Multiagent system, Evolutionary algorithm, Genetic algorithm, Natural computation, Adaptation, Evolutionary learning, Cooperative problem solving, Intelligent agent, Emergence, Complex system, Evolutionary system

1. **Introduction.** "Intelligent" systems (usually agent-based) are used in many domains of sciences. The development of the next generation intelligent systems (more intelligent than the actually developed) is an important research direction. Highly intelligent systems will be inherently complex having many interacting components (sometimes hybrid components) that require a very long time for their development. In the case of many very complex systems all the necessary data, information and knowledge for their development are obviously unavailable. Another aspect consists in the fact that it is necessary to elaborate more consecutive versions. Even if an intelligent system has some autonomy in increasing its intelligence sometimes by adaptation, there exists a point when the system is unable to make other improvement by itself and it is necessary a human intervention for the increasing of the system's intelligence.

Intelligent agent-based systems represent one of the most important approaches used for autonomous, efficient and flexible solving of difficult problems (tasks) and/or large

numbers of simple problems in many domains [7, 20, 44]. The main motivation consists in the properties of the agents that differentiate them from other computable systems. Agent-based systems can be endowed with capacities that allow to intelligently process computational hard (difficult/complex) problems [12, 19, 25]. There are many applications of the intelligent agents in many domains, including health care [18, 26, 32, 33, 34], which extend traditional developments. Recent implementations include applications, like *telehealth* [43], *analysis of spread simulation of infectious disease* [46], *web-enabled healthcare computing* [8], *patients monitoring* [17], *patients management* [30, 31], *medical diagnosis* [44] and *ubiquitous healthcare* [29].

In this paper, an adaptive multiagent system called *ERMS* (*Extended Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity*) is proposed. *ERMS* represents an extension, in order to operate in case of larger numbers of agents, of the *CCER* multiagent system (*Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity*) developed during our previous researches [23]. The results obtained during the development of *CCER* and *ERMS* multiagent systems prove that evolutionary learning algorithms can be successfully used by multiagent systems to establish how to adapt a more efficient problem-solving strategy when the emergent patterns of reorganization of the systems' structure can be discovered.

The upcoming part of the paper is organized as follows. Section 2 analyzes some aspects related with the intelligent adaptive systems. In Section 3 the novel adaptive multiagent system called *ERMS* is described – the evolutionary learning algorithm used by the *ERMS* system in order to learn different reorganizations is presented. At the end of Section 3 the correctness and efficiency in operation of the *ERMS* system is analyzed. In Section 4, *ERMS* System Intelligence is considered. In Section 5, the conclusions of the research are presented.

2. **Intelligent Adaptive Systems.** Many difficult problems solving require a specific sort of computational intelligence (capacities to intelligently handle the difficulties of the problems-solving) [12, 45]. In general a problem is considered difficult (computational hard) based on considerations such as the solving requires a large amount of resources (some times distributed resources); the solving requires a large quantity of problem solving knowledge (some times distributed knowledge); the solving requires a large variety of problem solving knowledge; the problem description contains different types of uncertainties (missing or erroneous data), etc. Intelligent agents must precisely and flexibly handle the solving of difficult problems or solving large amounts of difficult or simple problems. Cooperative agents can form multiagent systems, in that they can collaborate during the problems solving in order to make easier their solving and to improve the accuracy of the obtained problems solutions [12, 19, 20, 45, 47].

Many times it is difficult or even impossible to endow an agent with the necessary knowledge at the moment of its creation. Motivations may consist in considerations, like: some information initially is not known and/or some information are changing in time. These reasons motivate the necessity of endowment of the agents with autonomous learning capability [12, 19]. The purpose of learning consists in improvements in the efficiency and accuracy of the problems-solving.

In literature there are many adaptive multiagent systems [3, 4, 9, 10, 11, 15, 16, 23, 27, 28, 38, 39]. The adaptation many times is realized by leaning. One type of adaptation of a multiagent system consists in the reorganization of the problem-solving resources, in order to solve more efficiently the undertaken problems [10, 11, 23]. In a multiagent system the agents members of the system are called *problem-solving resources* (the agents solve the undertaken problems by the system) [10, 11]. An agent uses different resources (i.e.,

processing power, memory) during its life cycle for the problems solving. The significance of adaptability of the multiagent systems described in the papers [10, 11] is to allocate the resources that are necessary for the problems-solving. The adaptation in a multiagent system is realized at the system's level.

The papers [3, 15, 28] present some adaptive multiagent systems that efficiently use the available resources in the problems-solving process. In order to satisfy either the surplus or lack of resources, the presented multiagent systems self-organize depending on the necessities. The agents are endowed with capacity of *decomposition* and *composition*. The decomposition represents the capacity of an agent to create an identical copy with itself. The copy of an agent can solve the same set of problems like the initial agent (has the same problem-solving specializations). By composition, two agents are combined to become one in order to free computational resources. An agent obtained as a result of the composition of more agents has all the specializations of the agents used in the composition.

*TRACE* [10, 11] is an adaptive multiagent system, formed from coalitions of agents. The adaptability of the *TRACE* system consists in the capacity of each coalition to buy or sell resources (agents). Within the frame of the *TRACE* multiagent system, the resources are distributed in coalitions. The distribution is realized, depending on the necessities or on the surplus of resources within each of the coalitions. A coalition desires to buy resources, if it does not detain all the necessary resources for solving of problems. A coalition can sell resources if it does not need all the resources.

## 3. ERMS Multiagent System Description.

### 3.1. Previous researches.

*CCER* is an adaptive coalition-based multiagent system developed during our previous researches [23]. The adaptability of the system consists in the capacity to reorganize its structure depending on the received problems for solving. For the establishment of the reorganization an evolutionary learning algorithm is used. For the problems allocation for solving, a novel allocation protocol is used that represents an adaptation of a centralized problem allocation protocol as described in [12, 45].

The main advantage of the *CCER* multiagent system consists in the capacity to adapt its structure in order to solve more efficiently the undertaken problems. Simulation results show the increased efficiency of the *CCER* multiagent system when larger numbers of problems are transmitted for solving [23]. *CCER* system can reorganize its structure at the beginning of a problems solving cycle if some specific information, called *problems pattern* about the problems transmitted for solving exist. A problems solving cycle begins at the undertaking of a set of problems, and is finished when all the problems are solved. The establishment of reorganization requires a polynomial complexity search in a rule base. Thus, in the verification of the preconditions of some rules from the rule base, the identified rule is fired, neglecting the rest of the rules.

### 3.2. ERMS multiagent system architecture.

Each agent member of an intelligent cooperative multiagent system must have a *role*. A role defines the manner in which the agents who undertake that role contribute to the problems solving in the multiagent system [12, 19]. An agent who undertakes a role must have a set of specializations and necessary resources which allows to the agent to fulfill its role. A *multiagent system architecture* specifies different generic information [12, 45] (i.e., existent roles, relations between the roles, organization of the agents, specific cooperative problems solving methods used in the frame of the system, etc.) about the multiagent systems endowed with that architecture. The information that must describe a multiagent system architecture depends on conditions, like: complexity of the system, number of member agents, the

problems that have to be solved, the cooperative problem solving methods that can be used by the multiagent system, etc.

In the following, we propose a novel multiagent system architecture called *ERMS* (*Extended Centralized Multiagent System with Cooperative Evolutionary Reorganization Capacity*). We call *ERMS* a multiagent system with the *ERMS* architecture. The proposed architecture defines: a partially centralized multiagent system organization; a specific adaptation based on the reorganization of the system; a specific evolutionary learning algorithm; a specific cooperative problems-solving and the roles that can be undertaken by the agents. We call a problems pattern the description, values of different parameters related to a set of problems transmitted for solving, which may consists in information, like: what type of problems are transmitted for solving, the number of problems transmitted for solving.

In an *ERMS* multiagent system there are defined the following roles (see Figure 1): by centralized problem allocation for solving denoted *supervisor*; by problem-solving denoted *contractor*; by cooperative problem-solving (problem solving and local problem allocation for solving) denoted *manager*. An agent who undertakes the *contractor* role during the fulfilling of the role will solve problems. An agent who undertakes the *manager* role during the fulfilling of the role will solve problems and allocate problems for solving to some agents members of the system. An agent who undertakes the *supervisor* role during its life cycle will allocate problems for solving to other agents. In the multiagent system there is only one agent with supervisor role. The agent initially established with the supervisor role is not changing its role during the multiagent system's life cycle.

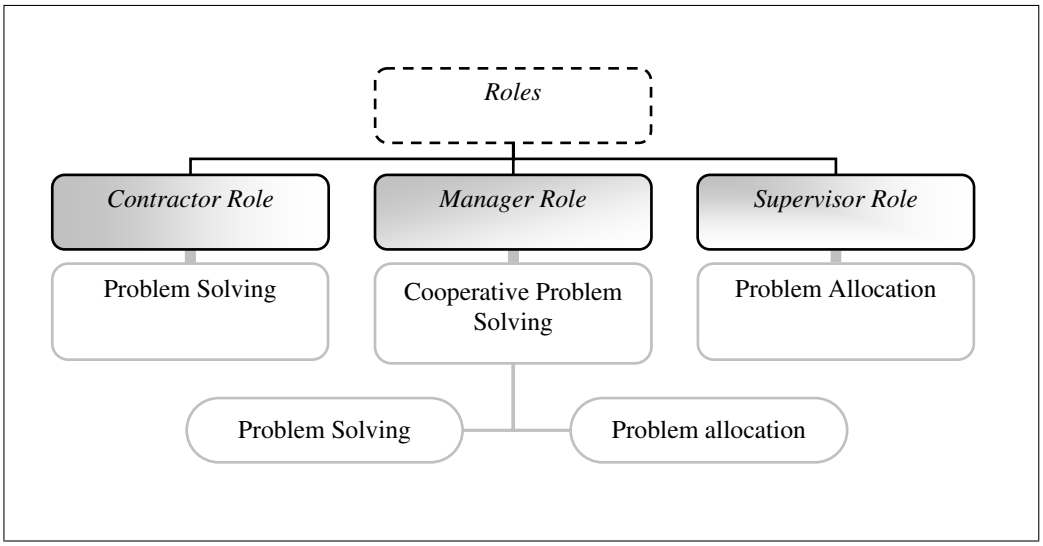



FIGURE 1. The roles in the ERMS multiagent system

An *ERMS* multiagent system at a problems-solving cycle is composed from a set $Ms$ (1) of agents, each of them having as unique identifier a natural number that is not changed during its life cycle. Each agent, except the supervisor, is associated one of the following abbreviations: "$Ct$" for *contractor agents* (for example, the notation $Ct_i$ has as meaning the agent with the identifier $i$ has contractor role), "$Mg$" for *manager agents* (for example, the notation $Mg_j$ has as meaning the agent with the identifier $j$ has manager role) and "$Fr$" for *free agents* (for example, the notation $Fr_k$ has as meaning the agent with the identifier $k$ has contractor role and it operates as free agent) that illustrate the specific of the agents contribution to the problems-solving. The agents, except the supervisor, can change their role during the system's operation. For example, an agent at

a problems-solving cycle may operate as contractor and at another problem solving cycle as manager.

$$Ms = Co \cup \{Su\} \cup Fr. \tag{1}$$

$Fr$ denotes a set of agents called free agents. A free agent $Fr_h$ ($Fr_h \in Fr$, $ID(Fr_h) = h$) has contractor role, $Role(Fr_h) = contractor$. A free agent does not belong to any coalition. $Su$ is the agent with the supervisor role, $Role(Su) = supervisor$. $Co$ represents the set of coalitions of agents.

A coalition of agents is composed from one or more agents with *manager* role and usually more agents with *contractor* role. In a coalition of agents each agent with contractor role is subordinated to a single agent with manager role. Figure 2 presents a coalition of agents denoted $Co_q$. $Mg_b$ ($Mg_b \in Co_q$, $role(Mg_b) = manager$, $ID(Mg_b) = b$) and $Mg_c$ ($Mg_c \in Co_q$, $role(Mg_c) = manager$, $ID(Mg_c) = c$) represent the managers of the coalition. $Ct_1, Ct_2, \ldots, Ct_x$ ($\{Ct_1, Ct_2, \ldots, Ct_x\} \subset Co_q$) represent the contractors subordinated to $Mg_b$. $Ct_a, Ct_b, \ldots, Ct_m$ ($\{Ct_a, Ct_b, \ldots, Ct_m\} \subset Co_q$) represent the contractors subordinated to $Mg_c$. $P_f, \ldots, P_v$ represent the problems that must be solved by $Mg_b$ and the subordinated agents. $P_g, \ldots, P_z$ represent the problems that must be solved by $Mg_c$ and its subordinated agents. The dashed arrows between the agents used in Figure 2, illustrate the communication and cooperation links between the agents. $t_a$ indicates a link by cooperation type (problem allocation for solving) between the agents.

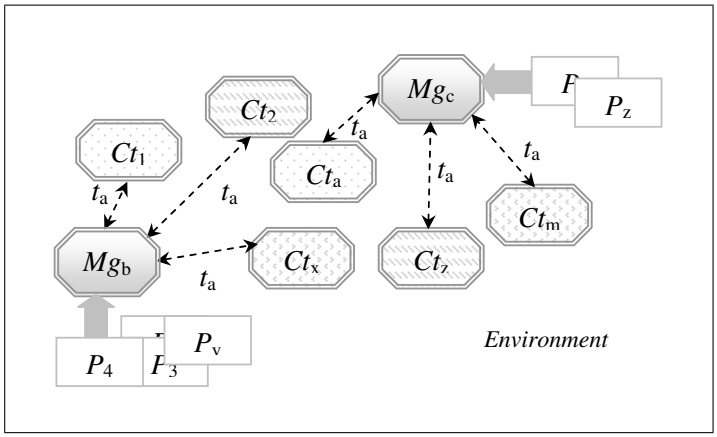



FIGURE 2. A coalition $Co_q$ of agents in the ERMS system

Figure 3 presents an *ERMS* multiagent system at a problems solving cycle, composed from the disjointed coalitions $Co = \{Co_1, Co_2, \ldots, Co_n\}$ ($\forall i \neq k, Co_i \in Co, Co_k \in Co, Co_i \cap Co_k = \varnothing$), the supervisor agent $Su$ and the free agents $Fr = \{Fr_1, Fr_2, \ldots, Fr_m\}$. $P = \{P_1, P_2, \ldots, P_y\}$ represents the problems transmitted for solving to $Su$. The dashed arrows used in Figure 3 illustrate the cooperation links in the multiagent system. $t_b$ indicates a link by cooperation type between the supervisor agent and a free agent. $t_c$ indicates a links by cooperation type between the supervisor agent and a coalition of agents.

An agent $Ct_k$ with contractor role ($role(Ct_k) = contractor$, $ID(Ct_k) = k$) is endowed with a specialization set $Spec(Ct_k) = \{S_1, S_2, \ldots, S_r\}$, which allows the solving of problems from a set $Cl = \{Cl_1, Cl_2, \ldots, Cl_r\}$ of classes of problems; where $S_i$ represents the specialization necessary for solving of the problems from the class $Cl_i$ of problems ($\forall i = \overline{1, r}, S_i \rightarrow Cl_i$).

An agent $Mg_k$ with manager role ($role(Mg_k) = manager$, $ID(Mg_k) = k$) is endowed with a specialization set $Spec(Mg_k) = \{S_1, S_2, \ldots, S_r\} \cup \{A_1, A_2, \ldots, A_e\}$. $S_1, S_2, \ldots, S_r$

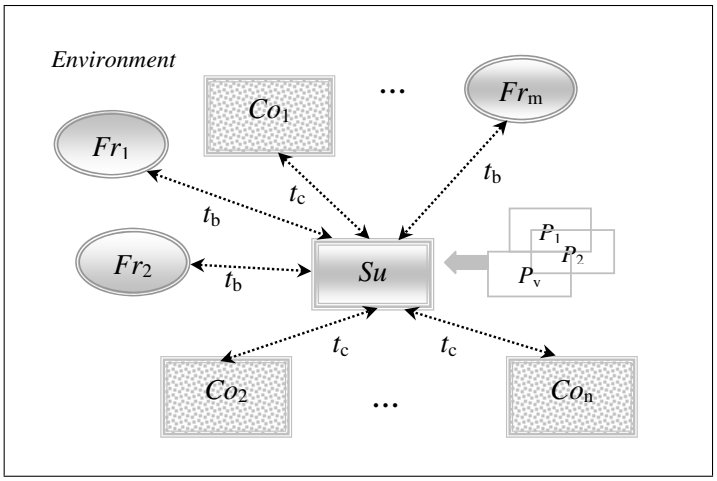



FIGURE 3. ERMS multiagent system at a problems solving cycle

allows the solving of the problems from a set $Cl = \{Cl_1, Cl_2, \ldots, Cl_r\}$ of classes of problems; where $S_i$ represents the specialization necessary for solving of the problems from the class $Cl_i$ of problems. If $Mg_k$ is manager in the coalition $Co_y$, then $A_1, A_2, \ldots, A_e$ represent knowledge detained by $Mg_k$ about the subordinated contractor agents from $Co_y$.

As examples of information detained by a manager agent $Mg_m$ from a coalition $Co_u$ ($Mg_m \in Co_u; role(Mg_m) = manager; ID(Mg_m) = m$), about the subordinated contractor agents, we mention: the number of contractor agents from $Co_u$; the specializations of each contractor agent; the information detained about the problems that are undertaken for solving by each contractor agent.

$Su$ detains information about the coalitions of agents and the free agents. As examples of information detained by $Su$ about a free agent $Fr_p$ ($role(Fr_p) = contractor$, $ID(Fr_p) = p$), we mention: $Fr_p$ specializations and capacity; the problems that are currently solved by $Fr_p$. As examples of information detained by $Su$ about a coalition $Co_r$ ($Co_r \in Co$), we mention: the number of member agents of $Co_r$; the capacity of the contractor agents members of $Co_r$; the problems that are currently solved by $Co_r$. The problems are transmitted for solving to $Co_r$ by $Su$. Each manager from $Co_r$ transmits the obtained problems solutions to $Su$. However, $Su$ knows when in the $Co_r$ coalition is finished an undertaken problem-solving.

3.3. **The ERMS system operation.** The *ERMS* multiagent system represents an extension of the *CCER* multiagent system [23]. One of the improvements consists in the use of coalitions with more manager agents, each of them with a set of subordinated contractor agents. This improvement was realized in order to eliminate the excessive centralized architecture of the *CCER* multiagent system. The centralization in a large-scale multiagent system may be a bottleneck in the operation of the system. Another adaptation consists in the use of genetic problem solving specializations (problem solving methods based on genetic algorithms).

Figure 4 illustrates a single problem-solving cycle in the *ERMS* multiagent system. A $P_h$ problem-solving begins at its undertaking and is finished when its solution $So_h$ is obtained. $Fr_v$ ($Fr_v \in Fr$, $role(Fr_v) = contractor$, $ID(Fr_v) = v$) represents a free agent. $Co_a$ ($Co_a \in Co$) represents a coalition of agents. $Mg_y$ ($Mg_y \in Co_a$, $role(Mg_y) = manager$, $ID(Mg_y) = y$) a manager in $Co_a$. $Ct_s$ ($Ct_s \in Co$, $role(Ct_s) = contractor$, $ID(Ct_s) = s$) is a contractor agent in $Co_a$ subordinated to $Mg_y$.

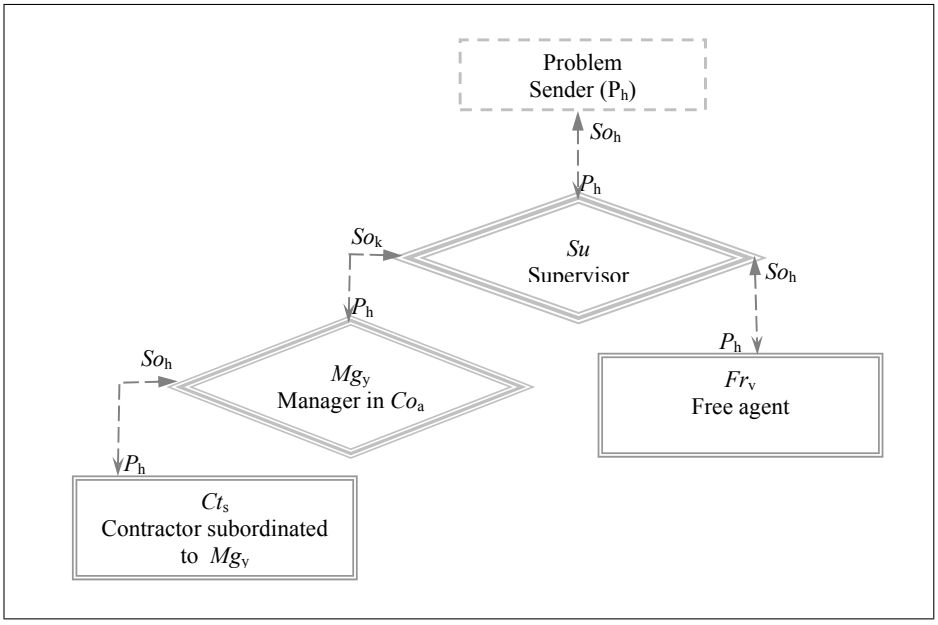



FIGURE 4. A problem's solving cycle in the ERMS multiagent system

In the *ERMS* multiagent system, each coalition and free agent can solve problems transmitted for solving by the supervisor agent. In a coalition, each transmitted problem is received by a manager from the coalition. In the case of a received problem, the manager agent will solve the problem or will transmit it for solving to a subordinated contractor agent. The manager agents from the coalitions are responsible for the problems solutions undertaking from the subordinated contractor agents and their transmission to the supervisor agent.

A problem denoted $P_h$ transmitted for solving has the description $Desc_h$ (2).

$$Desc_h : \langle sender_h, f_h, type_h, priority_h, S_q \rangle. \tag{2}$$

In (2) there are used the following notations: $sender_h$ specifies the sender of $P_h$; $type_h$ represents the type (minimization or maximization) of $P_h$; $f_h$ represents the objective function that must be minimized or maximized; $priority_h$ represents the priority of the problem; $S_q$ represents the specification of the specialization to be used for the problem-solving. It is always solved a maximization problem, a received minimization problem is transformed into a maximization one.

To a specialization $S_q$ is associated a set of parameters called $pr_q$ (3) having an associated identifier $in_q$.

$$(pr_q, in_q) : \langle se_q, oi_q, om_q, cd_q, ln_q, nc_q, n_q, pc_q, pm_q \rangle. \tag{3}$$

$se_q$ specifies the used selection operator (for example, is used the Monte Carlo linear selection method, $se_q$=*"Monte Carlo"*). $cd_q$ describes the encoding of the genes from the chromosomes (for example, the genes are binary encoded, $cd_q \in \{0, 1\}$). $ln_q$ represents the length of the chromosomes (for example, $ln_q = 20$ denotes chromosomes composed from 20 genes). Each chromosome obtained during a problem-solving process has the same length. $nc_q$ represents the number of chromosomes from a generation (for example, $nc_q = 50$). Each generation has the same number of chromosomes. $n_q$ represents the number of the created generations of chromosomes during a problem-solving. $oi_q$ specifies the used crossover operator (with a single crossover point, for example). $om_q$ specifies the used mutation operator. $pc_q$ represents the probability of crossover. $pm_q$ specifies the probability of mutation.

In (4) is presented a genetic problem solving specialization denoted $S_q$.

$$(S_q, id_q) : \langle in_q, out_q, type_q, method_q \rangle. \tag{4}$$

In (4) there are used the following notations: $S_q$ specifies how the specialization is called (to each specialization is associated a name); $id_q$ denotes $S_q$ identifier; $in_q$ denotes the identifier associated to the input parameters for $S_q$; $out_q$ denotes the output parameter for the $S_q$ specialization (the problem solution obtained after its solving); $type_q$ represents the type minimization or maximization; $method_q$ contains the description of the problem-solving method. A specialization is defined by a problem-solving method, the problem type and the input parameter values with that are initialized the parameters of the genetic algorithm.

An agent with contractor role receives problems specified in the form (2). The agent based on the problem description establishes the necessary problem-solving specialization by the form (4). The result obtained after the running of the genetic method described in the specialization represents the problem solution. The solving of a problem using a genetic algorithm does not supposes the interpretation of a syntactically described code. However, it does not decreased the problem-solving time. The manager agents can solve problems like the contractor agents. A manager agent must have specializations, which allows problems allocation for solving to contractor agents. A manager agent must distribute a problem to a contractor agent if it does not have the necessary capacity to solve the problem.

A problem-solving specialization does not represent a code difficult to use in the problems-solving. The form (4) can be adapted for problems solving using different problem solving methods based on evolutionary algorithms (evolutionary programming, genetic programming, evolutionary strategies, etc.).

3.4. **The adaptivity of the ERMS multiagent system.** In the following, we consider an *ERMS* multiagent system, denoted *ASE*. The problems transmitted for solving at a problems-solving cycle may match a *problems pattern*. A problems pattern describes different information related with the transmitted problems for solving. Formula (5) presents the general form of a problems pattern denoted $Pat_i$ specified by the specializations denoted $S_1, S_2, \ldots, S_n$ used in the problems-solving; numbers of problems denoted $nr_1, nr_2, \ldots, nr_n$ by each specialization and the associated priorities denoted $priority_1, priority_2, \ldots, priority_n$.

$$Pat_i : \langle S_1, nr_1, priority_1; S_2, nr_2, priority_2; \ldots; S_n, nr_n, priority_n \rangle \tag{5}$$

*ASE* system reorganization is described by a set $Rl$ (6) of rules detained by $Su$ in a rule base.

$$Rl = \{Rl_1, Rl_2, \ldots, Rl_k\}. \tag{6}$$

An $Rl_i$ ($Rl_i \in Rl$) rule has the form (7).

$$(Rl_i, Id_i) : \langle Pat_i \rangle \rightarrow \langle Inst_i \rangle. \tag{7}$$

$Id_i$ represents the $Rl_i$ rule identifier (each rule has as identifier a unique natural number). $Pat_i$ represents a problems pattern. $Inst_i$ defines an instance of the *ERMS* multiagent system architecture, distribution of the agents in coalitions and allocation of roles to the agents.

*ASE* system can reorganize autonomously its structure at the beginning of each problems-solving cycle, if the problems pattern is known at the beginning of the problems-solving cycle, and $Su$ has a rule in its rule base whose precondition matches the problems pattern. A reorganization determination implies a search by $Su$ in its set $Rl = \{Rl_1, Rl_2, \ldots, Rl_k\}$ (6) of rules. The rule whose precondition matches the known problems

pattern is selected. The postcondition of the selected rule defines the new instantiation of the *ASE* system. $Su$, after establish the multiagent system instance, announces each agent to which coalition must migrate and what role must undertake or must operate as a free agent. Each agent will migrate autonomously into the coalition which must belong or will operate as a free agent. The determination of a new instance of the *ASE* system has a polynomial complexity.

Fundamental aspects of the *evolutionary algorithms* has been analyzed by many authors [1, 5, 13, 20]. *Evolutionary learning techniques* that are based on methods of evolutionary computation [23, 24], represent a subclass of learning techniques [12, 19]. In the following, an evolutionary learning algorithm called *Evolutionary Reorganization Learning*, which allows the construction of a rule by the form (7), denoted $Rl_i$ is described. Each rule detained by $Su$ will be created using the learning algorithm. *ASE* system has *no+1* member agents (a supervisor agent and *no* agents that can operate in the frame of the coalitions undertake *contractor* or *manager* roles or can operate as free agents with contractor role). Each agent, except the supervisor agents is attached a unique identifier (in the system does not exists two agents with the same identifier).

*Algorithm – Evolutionary Reorganization Learning*

{$In$: $Pat_i$ – a problems pattern}

{$Out$: $Rl_i$ – the constructed rule}

*Step 1. The creation of the initial generation of chromosomes.*

$t = 1$.

@Initialize the chromosome population $P(t)$.

*Step 2. Search for the best-fitted instantiation of the ASE architecture.*

*While* $(t \leq gen)$ *do*

@Create a copy $Cr_y$ of the best-fitted chromosome from $P(t)$.

@Select chromosomes from $P(t)$ using a selection method. Choose chromosomes from $P(t)$ to enter in the mating pool *mp*. Let $P_1$ be the selected chromosomes.

@Using the *rc* crossover operator applied with the probability *pr* recombine the chromosomes in *mp* forming the population $P_2$.

@Mutate the chromosomes in $P_2$ using the operator *mc*, with the probability *pm*.

@Replace in $P_2$ the worst-fitted chromosome with $Cr_y$. Let $P_2$ be the obtained population of chromosomes.

$t = t + 1$.

$P(t) = P_2$ (the new generation of chromosomes is constructed).

*EndWhile*.

*Step 4. Construction of the rule that describes the ASE system's organization.*

@Select the best-fitted chromosome $Cr_y$ from $P(t)$.

@Generate a unique rule identifier denoted $Id_i$.

@Based on the selected chromosome $Cr_y$ construct the $Rl_i$ rule.

$(Rl_i, Id_i) : \langle Pat_i \rangle \rightarrow \langle Inst_i \rangle$.

*EndEvolutionaryReorganizationLearning*.

$gen, n, pm, pr, no$ are parameters of the algorithm. $gen$ represents the number of generations of chromosomes constructed during a learning process. Let us denote with $P(t)$ the $t$'th generation of chromosomes. In $P(1)$ each chromosome is generated by random. Each generation has the same number $n$ of chromosomes. Each chromosome has $no$ genes. A chromosome $Cr_w = [G_1, G_2, \ldots, G_{no}]$ specifies an instance of the $ASE$ architecture.

Except $Su$, to each agent $Ag_j$ ($ID(Ag_j) = j$) a gene $G_j = (v_j[1], t_j[2], w_j[3])$ corresponds in each chromosome. The first layer parameter value $v_j$ from the gene $G_j$ specifies the affiliation of $Ag_j$ to the coalition with the identifier $v_j$. Each coalition is identified with a natural number $v_k$, where $v_k \in [1, no]$. All the agents with the same value $v_k$ associated to their corresponding gene are members of the same coalition, the coalition with the identifier $v_k$. A gene that has a value different from all other genes' value means that the corresponding agent to the gene is a free agent. The second layer value $t_j$ ($t_j \in T$, where $T = \{'m', 'c'\}$) from the gene $G_j$ specifies the role of the $Ag_j$ agent. If $t_j =' m'$ then $role(Ag_j) = manager$. If $t_j =' c'$ then $role(Ag_j) = contractor$. If $t_j =' c'$ and $Ag_j$ is not a free agent, then $w_j$ value specifies the identifier of the manager agent from the same coalition to who will be subordinated $Ag_j$. A free agent, denoted $Ag_j$, has contractor role ($role(Ag_j) = contractor$, $t_j =' c'$) and $w_j = 0$ (does not subordinated to any manager agent). An offspring generation is created using specific selection, crossover and mutation operators.

By mutation, denoted $mc$, are generated new chromosomes by small variations of the genes' values in the chromosomes. We define $mc$ as an application by the form (8). $Cr$ specifies the chromosome space.

$$mc : Cr \to Cr. \tag{8}$$

The mutation is applied on each layer of each gene $G_j = (v_j[1], t_j[2], w_j[3])$ from each chromosome with the probability $pm$, $pm = \{pm_v, pm_t\}$, where $pm_v$, $pm_t$, are the corresponding probabilities to the mutation of $v_j$ and $t_j$. $pm_t$ represents the probability to be created a contractor agent that will operate in the frame of a coalition. $1 - pm_t$ represents the probability to be created a manager agent, where $pm_t > 1 - pm_t$. By mutation, the $v_j$ value may increase or decrease, the new value of $v_j$ must be between 1 and $no$.

If $v_j$ specifies that the agent is a free agent then there is not applied the mutation to the rest of the layers. $t_j$ is set to $'c'$ ($t_j =' c'$ – the agent will operate as having contractor role) and $w_j$ is set to 0 ($w_j = 0$ – the agent is not subordinated to any manager agent).

If $v_j$ does not specify a free agent then there is applied the mutation to the second layer $t_j$ that could change its value, the new value of $t_j$ must be in the set $T = \{'m', 'c'\}$, where $t_j \in T$. If $t_j = 'm'$ then $w_j$ value is set to 0. If $t_j = 'c'$ then is randomly generated one of the identifiers of manager agents from the same coalition and the identifier is set to $w_j$.

In case of each coalition is verified if all the managers of the coalition have subordinated contractor agents. The role of a manager that does not have subordinated contractor agents is changed into contractor and the agent is subordinated randomly to a manager agent from the same coalition that have at least on other contractor.

The crossover $rc$ is used to create new chromosomes by combining the genetic information of parent chromosomes. We define $rc$ as an application by the form (9).

$$rc : Cr^2 \to Cr^2. \tag{9}$$

$rc$ realizes a (2, 2) transformation, two parents are combined to obtain two offspring. $rc$ is applied with the probability $\{pr; \{pr_v, pr_t, pr_w\}\}$; where $pr$ represents the probability of the chromosomes to be selected for inclusion in the mating pool, $pr_v, pr_t, pr_w$ represent probabilities to crossover the layers 1, 2 and 3 in the chromosomes. During the crossover of two chromosomes there are recombined the values of the genes from the same layer: [1]

for the $v_i$ values (that specify the agents membership to coalitions); [2] for the $t_i$ values (that specify the agents roles) and [3] for the $w_i$ values (if an agent is contractor then $w_i$ specify its manager agent to who the contractor agent is subordinated).

Each chromosome of a population is evaluated using a real-valued fitness function $Fit$ by the form (10), where $\forall Cr_b \in Cr, Fit(Cr_b) \geq 0$, which counts how efficiently the multiagent system with the structure specified by the chromosome $Cr_b$ can solve the undertaken problems at a problems-solving cycle.

$$Fit : Cr \to R^+. \tag{10}$$

A chromosome's fitness is evaluated simulating the problems-solving that matches the problems pattern. The efficiency of the problems-solving at a problems solving cycle, has as meaning the problems solving time (all the undertaken problems at the beginning of the problems-solving cycle are solved). Let $Cr_h$ ($Cr_h \in Cr$) and $Cr_k$ ($Cr_k \in Cr$) two chromosomes. $Fit(Cr_h) > Fit(Cr_k)$ means that $Cr_h$ is best-fitted then $Cr_k$ (it is solved a maximization problem, an initially transmitted minimization problem is transformed to a maximization one). The best-fitted chromosome $Cr_y$ from the least generation $P(gen)$ specifies the postcondition of the constructed $Rl_i$ rule.

For each gene $G_j = (v_j, t_j, w_j)$ from a chromosome $Cr_w$ must be satisfied the following validity restrictions:

**A)**. Let $v_j = nr_j$, then $nr_j \in [1, no]$ and must have the values 1,..., $nr_j - 1$ in the first layer of the $Cr_w$ chromosome;

**B)**. Let $t_j = tr_j$, then $tr_j \in \{'m', 'c'\}$;

**C)**. Let $w_j = wr_j$. If $tr_j =' c'$ and $nr_j$ does not have a unique value (it does not specify a free agent), then $wr_j$ must specify the identifier of a manager agent from the same coalition (coalition with the identifier $nr_j$) as the agent with the identifier $j$. If $tr_j =' m'$ then $wr_j = 0$. If $tr_j =' c'$ and $nr_j$ has a unique value (specify a free agent), then $wr_j = 0$.

**D)**. Let $t_j = tr_j$. If $tr_j =' m'$ then it must have at least one subordinated contractor agent.

In the case of each invalid chromosome obtained during a learning process (are not satisfied all the restrictions $(A)$, $(B)$, $(C)$ and $(D)$) a transformation $Trf$ (11) is applied that corrects the invalid genes values.

$$Trf : Cr \to Cr. \tag{11}$$

The validity of each randomly generated chromosome from the initial generation is verified. In case of an invalid chromosome is applied the transformation $Trf$ (11). During a learning process, the validity of each newly obtained chromosome by applying the mutation $mc$ or crossover $rc$ is verified. A chromosome is considered valid if it specifies a correct multiagent system structure.

A survival mechanism based on the fitness measure $Fit$ is applied to select the chromosomes of the new generation from the offspring and parent generations. In the *Evolutionary Reorganization Learning* algorithm two types of selections are used. The selection for crossover operator is used to decide which members of the recent generation $P(t)$ will be used as parents of the new generation $P(t+1)$. The selection for the replacement operator is used to obtain, which chromosomes from $P(t)$ and their offspring will effectively enter in the new generation $P(t+1)$. The best-fitted chromosome $Cr_z$ from the last generation $P(gen)$ represents the solution (a valid multiagent system structure).

3.5. **Correctness in operation of the ERMS system.** During its operation for the reorganization, the *ERMS* system uses a set of learned rules. All the rules are correct, their postcondition specify correct reorganization of the *ERMS* system. The correctness of the *Evolutionary Reorganization Learning* algorithm can be theoretically demonstrated.

Each chromosome obtained during a learning process, represents a valid instantiation of the *ERMS* multiagent system architecture. Only the $mc$ (8) mutation and $rc$ (9) crossover operators can modify the genes values from the chromosomes. A transformation $Trf$ (11) is applied to each obtained invalid chromosome. However, at the end of each learning process a correct multiagent system instantiation is obtained. In every new generation, the best-fitted chromosome from the previous generation is transferred, which guarantees that the best multiagent system instantiation is not lost during the learning process.

The *ERMS* system can reorganize its coalitions at the beginning of each problems-solving cycle, if there is known at the beginning of the problems-solving cycle the problems pattern and there is a rule whose precondition match that pattern. In case of selection of a rule at the beginning of a problems pattern, the postcondition of that rule specifies the necessary correct reorganization of the system. If it is not matched any rule at the beginning of a problems solving cycle then the system remains with the previous structure.

For the validation of the *ERMS* multiagent system, there have been realized experimental simulations for the learning processes and the testing of the system's operation with some learned rules by sending to the system problems at the beginning of problem-solving cycles with known and unknown pattern. The simulations purposes were to establish the measure in that the distribution of agents in coalitions and allocation of roles to the agents, influences the efficiency of the problems-solving. During the simulations, they used agents endowed between 1 and 16 problem-solving specializations (*a specialization* is a problem-solving method where some parameters could be initialized during a problem-solving). Requirements for an agent to solve a problem is to have the necessary role (role that allows problems-solving), the necessary problem-solving specialization and resources.

We have simulated learning processes were generated at least $gen = 57$ generations of chromosomes. The problems have been solved in problems-solving cycles. At each problems-solving cycle the problems have been transmitted for solving at the beginning of the cycle. In the case of each problems pattern, there have been realized 50 simulations for the construction of the rule based on the problems pattern. The necessity to run multiple times learning processes on the same data was because it has been used an evolutionary learning technique (as in any heuristic search in the problem space running the same evolutionary algorithm on the same data with the same parameters many converge to different solutions). The most appropriate parameters values used during the simulations were: mutation probability $pm = \{pm_v, pm_t\} = \{0.0093, 0.8\}$; crossover probability $\{pr; \{pr_v, pr_t, pr_w\}\} = \{0.09; \{0.2, 0.3, 0.33\}\}$. There were made experiments for the following conditions: numbers of problems transmitted for solving (denoted $prno$): 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140 and 150; numbers of agents (denoted $no$): 10, 20, 25, 30, 35 and 40; chromosomes numbers (denoted $n$) in the generations (during a learning process each generation having the same number of chromosomes): 20, 25, 30 and 35.

Figures 5-8 present the changing of the average problem-solving time (expressed in $ms$ milliseconds), from generation to generation (each generation where composed from 30 chromosomes), for the first 29 generations of chromosomes using 25 agents, and 57 generations of chromosomes using 20 agents for the construction of 3 rules.

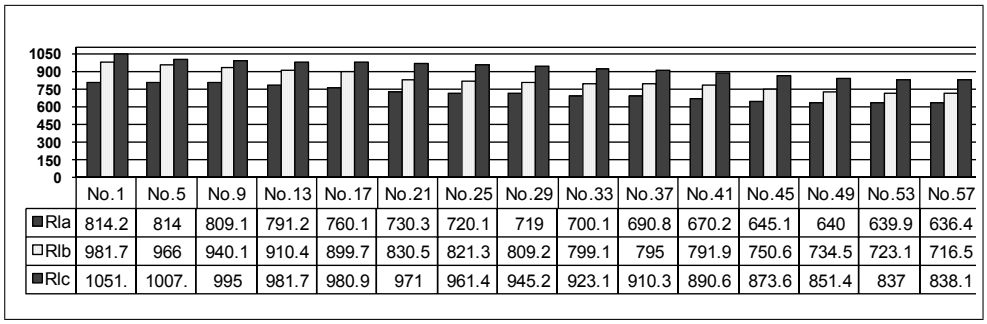
Figure 5 presents the time decrease using $no = 20$ agents, for the construction of 3 rules, based on 3 problems patterns, each of them composed from $prno = 70$ problems. The using of the rules has an improvement (as time decrease) of $\sim 28\%$ for $Rl_a$, $\sim 37\%$ for $Rl_b$, and $\sim 25\%$ for $Rl_c$. By using $no = 20$ agents that solve $prno = 70$ problems arranged in patterns the average improvement obtained during the simulations was approximatively by 29%.

Figure 6 presents the time decrease using $no = 20$ agents, for the construction of 3 rules, based on 3 problems patterns, each of them composed from $prno = 140$ problems. The using of the rules has an improvement of $\sim 20\%$ for $Rl_d$, $\sim 23\%$ for $Rl_e$, and $\sim 20\%$ for $Rl_f$. By using $no = 20$ agents that solve $prno = 140$ problems arranged in patterns the average improvement obtained during the simulations was approximatively by 22%.

Figure 7 presents the time decrease using $no = 25$ agents, for the construction of 3 rules, based on 3 problems patterns, each of them composed from $prno = 70$ problems. The using of the rules has an improvement of $\sim 30\%$ for $Rl_g$, $\sim 40\%$ for $Rl_h$, $\sim 20\%$ for $Rl_i$. By using $no = 25$ agents that solve $prno = 70$ problems arranged in patterns the average improvement obtained during the simulations was approximatively by 31%.

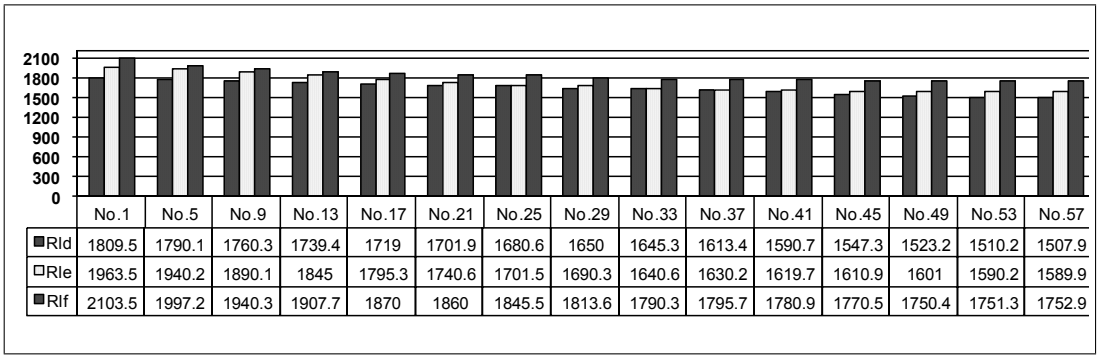
Figure 8 presents the time decrease using $no = 25$ agents, for the construction of 3 rules, based on 3 problems patterns, each of them composed from $prno = 140$ problems. The using of rules has an improvement of $\sim 21\%$ for $Rl_j$, $\sim 33\%$ for $Rl_k$, $\sim 19\%$ for $Rl_l$. By using $no = 25$ agents that solve $prno = 140$ problems arranged in patterns the average improvement obtained during the simulations was approximatively by 24%.

During a problems-solving at a problems solving cycle there are always some costs associated with the computation time. There is a cost for checking if the problems sent for solving respects a known pattern (there is a rule in the rule base that has as precondition the problems pattern which have been learned during a learning process). If the problems do not respect any pattern then there is a cost without any improvement. If at the beginning of a problems-solving cycle is verified a rule, then there is a cost for the checking of the reorganization.



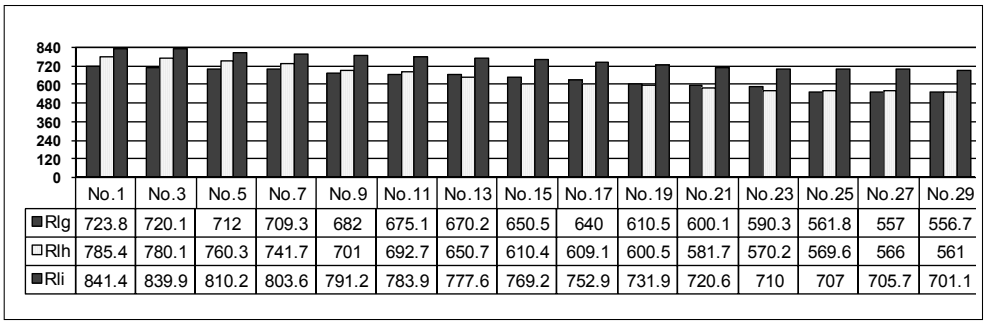

| | No.1 | No.5 | No.9 | No.13 | No.17 | No.21 | No.25 | No.29 | No.33 | No.37 | No.41 | No.45 | No.49 | No.53 | No.57 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rla | 814.2 | 814 | 809.1 | 791.2 | 760.1 | 730.3 | 720.1 | 719 | 700.1 | 690.8 | 670.2 | 645.1 | 640 | 639.9 | 636.4 |
| Rlb | 981.7 | 966 | 940.1 | 910.4 | 899.7 | 830.5 | 821.3 | 809.2 | 799.1 | 795 | 791.9 | 750.6 | 734.5 | 723.1 | 716.5 |
| Rlc | 1051. | 1007. | 995 | 981.7 | 980.9 | 971 | 961.4 | 945.2 | 923.1 | 910.3 | 890.6 | 873.6 | 851.4 | 837 | 838.1 |

FIGURE 5. Construction of the rules $Rl_a$, $Rl_b$, $Rl_c$, generations 1-57, $no = 20$, $prno = 70$, $n = 30$



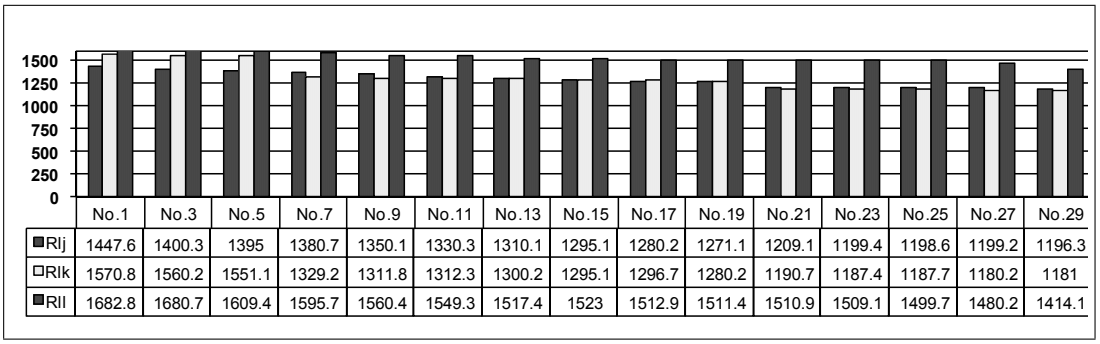

| | No.1 | No.5 | No.9 | No.13 | No.17 | No.21 | No.25 | No.29 | No.33 | No.37 | No.41 | No.45 | No.49 | No.53 | No.57 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rld | 1809.5 | 1790.1 | 1760.3 | 1739.4 | 1719 | 1701.9 | 1680.6 | 1650 | 1645.3 | 1613.4 | 1590.7 | 1547.3 | 1523.2 | 1510.2 | 1507.9 |
| Rle | 1963.5 | 1940.2 | 1890.1 | 1845 | 1795.3 | 1740.6 | 1701.5 | 1690.3 | 1640.6 | 1630.2 | 1619.7 | 1610.9 | 1601 | 1590.2 | 1589.9 |
| Rlf | 2103.5 | 1997.2 | 1940.3 | 1907.7 | 1870 | 1860 | 1845.5 | 1813.6 | 1790.3 | 1795.7 | 1780.9 | 1770.5 | 1750.4 | 1751.3 | 1752.9 |

FIGURE 6. Construction of the rules $Rl_d$, $Rl_e$, $Rl_f$, generations 1-57, $no = 20$, $prno = 140$, $n = 30$




| | No.1 | No.3 | No.5 | No.7 | No.9 | No.11 | No.13 | No.15 | No.17 | No.19 | No.21 | No.23 | No.25 | No.27 | No.29 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rlg | 723.8 | 720.1 | 712 | 709.3 | 682 | 675.1 | 670.2 | 650.5 | 640 | 610.5 | 600.1 | 590.3 | 561.8 | 557 | 556.7 |
| Rlh | 785.4 | 780.1 | 760.3 | 741.7 | 701 | 692.7 | 650.7 | 610.4 | 609.1 | 600.5 | 581.7 | 570.2 | 569.6 | 566 | 561 |
| Rli | 841.4 | 839.9 | 810.2 | 803.6 | 791.2 | 783.9 | 777.6 | 769.2 | 752.9 | 731.9 | 720.6 | 710 | 707 | 705.7 | 701.1 |

FIGURE 7. Construction of the rules $Rl_g$, $Rl_h$, $Rl_i$, generations 1-29, $no = 25$, $prno = 70$, $n = 30$




| | No.1 | No.3 | No.5 | No.7 | No.9 | No.11 | No.13 | No.15 | No.17 | No.19 | No.21 | No.23 | No.25 | No.27 | No.29 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rlj | 1447.6 | 1400.3 | 1395 | 1380.7 | 1350.1 | 1330.3 | 1310.1 | 1295.1 | 1280.2 | 1271.1 | 1209.1 | 1199.4 | 1198.6 | 1199.2 | 1196.3 |
| Rlk | 1570.8 | 1560.2 | 1551.1 | 1329.2 | 1311.8 | 1312.3 | 1300.2 | 1295.1 | 1296.7 | 1280.2 | 1190.7 | 1187.4 | 1187.7 | 1180.2 | 1181 |
| Rll | 1682.8 | 1680.7 | 1609.4 | 1595.7 | 1560.4 | 1549.3 | 1517.4 | 1523 | 1512.9 | 1511.4 | 1510.9 | 1509.1 | 1499.7 | 1480.2 | 1414.1 |

FIGURE 8. Construction of the rules $Rl_j$, $Rl_k$, $Rl_l$, generations 1-29, $no = 25$, $prno = 140$, $n = 30$

The testing of the system was realized using 25 agents (rules constructed for a system composed from 25 agents) by sending to it 60% known and 40% unknown problems patterns. The results of simulations show an average improvement of about 19.14% decrease in the solving time when the system reorganizes its structure versus the system does not (the average decrease of the solving time was 23.24%, with 4.1% reorganization costs). The optimal number of parameters related to the number of agents was 25. The most appropriate number of chromosomes in the simulation settings was 30. A smaller number of chromosomes have increased the number of necessary generations, increasing the total learning time. A larger number of chromosomes have not increased the convergence time, and after 29 and 57 generations it does not have significant improvement.

4. **Considerations Related with the Intelligence – ERMS System Intelligence.** In numerous studies are given definitions and hypotheses related with different types of *biological intelligence*, like *human intelligence* [14, 36, 37, 40], *animal intelligence* [2] and *plant intelligence* [41, 42]. Based on some recent studies [2, 14, 36, 37, 40, 41, 42], we can conclude that the biological intelligence in general, human intelligence particularly could not be defined unequally because it is not completely understandable. Some of the difficulties in its understanding consist in aspects like the biological complexity and variety. Human intelligence cannot be defined but may be measured from different points of view. The *psychometric* approach is by far the most widely used in practical settings for measurement of the human intelligence [36].

The humans have attained the actual level of intelligence during a very long evolution. As long as the intelligence of the humans and computational systems are completely different, they cannot be directly compared; therefore, the intelligence must be evaluated

based on different considerations. In many researches the biological intelligence represents a source of inspiration for the development of intelligent systems.

We consider the "intelligence" a property (or a set of properties) of a system (usually an agent) that emerges in some improvements in the problems-solving, many times consisting in increased autonomy, precision and flexibility of the problems-solving. Commonly the computational systems intelligence is considered just based on some properties, like capacity to: learn, adapt, evolve etc. Such considerations as they are only specified could not be considered rigorous for a system's intelligence assessment.

An agent must have only the necessary intelligence. Sometimes, unnecessary intelligence, usually in the case of solving very simple problems, may decrease the efficiency of the problems-solving. Usually, an intelligent agent makes some computations during the problems-solving that improves the efficiency and flexibility of difficult problems-solving. But such computations, in case of simple problems, became unnecessary and time consuming. The establishment of the necessary intelligence for an agent is an important aspect that must be analyzed at its development cycle.

For illustrative purposes, we consider as example the assessment of a system's intelligence based on the capacity to learn knowledge that allows new problems-solving. There are different aspects that must be taken into consideration at the evaluation of the intelligence:

- The learning time. The system can learn on-line or off-line;
- The quantity of learned knowledge. The system can learn more or less knowledge;
- The accuracy/quality of learned knowledge. The learned knowledge could be more or less accurate;
- The usefulness of the learned knowledge. The learned knowledge could be more or less useful. It could happen that the system will not use the learned knowledge;
- The consumption of computational resources during the learning. The learning could require more or less computational resources (could be more or less time consuming);
- The consumption of computational resources during the problems-solving. The use of the learned knowledge may require more or less computational resources. We may consider for example an extremely intelligent system that uses numerous resources for solving of a simple problem making useless computations.

Our consideration is that even if it is not possible to give a unified definition for the intelligence of a system in general, for the evaluation of a system's intelligence we consider necessary the following aspects to be established:

1. existence of one or more properties based on which the system could be considered intelligent. The intelligence is manifested by evolution (the system evolve autonomously), for example.
2. elaboration of a metric that allows the measurement of the intelligence (allows a quantitative evaluation of quality). The metric must indicate the existence of the intelligence. Sometimes it is better to indicate a degree of intelligence, like: *no intelligence*, *limited intelligence*, *normal intelligence*, *increased intelligence*, *extreme intelligence*.

A metric (general evaluator) that allows the measurement of the intelligence of a system must take into consideration aspects, related with the:

- specific and type of the system. For example, the system is a: *static software agent*; *mobile software agent*; *mobile robotic agent*; *static robotic agent*.
- specific, number and complexity of the problems that must be solved by the system. Usually a difficult problem solving requires more intelligence. Different types of problems could require different type of intelligence for their solving.

- the autonomy of the system in attaining the intelligence;
- the necessary cost and duration for attaining of intelligence;
- the measurable improvements that emerge based on the detained intelligence;
- measurable time in that the use of intelligence has as result improvements.

5. **Conclusions.** The main purpose of our research was a study of the intelligence that emerges in an adaptive system composed from relatively simple cooperating agents. We have proposed a multiagent system called *ERMS* capable of solving relatively large numbers of problems using genetic algorithms. Many real life problems-solving require the use of genetic algorithms or combinations of them with other methods [1, 5, 6, 13, 20]. Moreover, in the papers [21, 22] is described a novel class of mobile software agents called *ICMAE* (*Intelligent Cooperative Mobile Agents with Evolutionary Problem Solving Specialization*), that uses problem-solving specializations based on evolutionary problem-solving techniques.

The properties of the *ERMS* system that could be associated with the intelligence consist in the adaptability of the system, manifested by its capacity to autonomously reorganize its structure. The system is able to autonomously learn, using an evolutionary learning technique, how to adapt its structure. As a metric for the evaluation of the systems' intelligence we have considered the problems-solving time resulted from the systems' reorganization based on the specific/pattern of the problems sent for solving.

*ERMS* multiagent system represents an extension of the *CCER* multiagent system [23] developed during our previous researches. The development presented in this paper and the previously developed *CCER* multiagent system proves that computational system, composed from interacting components (agents), that use methods based on evolutionary computation for learning the required adaptability, exhibits at the level of the system an emergent intelligent adaptive behavior.

Usually, the intelligence of a system gave advantages in some situations, but it could have in some conditions disadvantages as well. We have established some general principles that must guide the development of intelligent systems and estimation of their intelligence. Usually a system's intelligence increases its complexity which the system must be able to handle autonomously inside. The complexity of a system must be hidden from the external parties, to the humans and agents that request problems-solving from the system.

**Acknowledgments.** The research of Barna Laszlo Iantovics was supported by the project "Transnational Network for Integrated Management of Postdoctoral Research in Communicating Sciences. Institutional building (postdoctoral school) and fellowships program (CommScie)" – POSDRU/89/1.5/S/63663, financed under the Sectoral Operational Programme Human Resources Development 2007-2013.

The research of Constantin-Bala Zamfirescu was supported by the research project between Romania and Slovakia, entitled *Hybrid Medical Complex Systems – ComplexMediSys*, 2011-2012. The partner institutions involved in the project are Petru Maior University, Tg. Mures, Romania and the Institute of Informatics of the Slovak Academy of Sciences, Bratislava, Slovakia.

## REFERENCES

[1] T. Back, D. B. Fogel and Z. Michalevitz, *Handbook of Evolutionary Computation*, Oxford University Press, Oxford, 1997.
[2] S. Coren, *The Intelligence of Dogs*, Bantam Books, 1995.
[3] D. D. Corkill, *A Framework for Organizational Self-Design in Distributed Problem Solving Networks*, University of Massachusetts, 1983.

[4] K. S. Decker, K. Sycara and M. Williamson, Cloning for intelligent adaptive information agents, *Lecture Notes in Computer Science*, vol.1286, pp.63-75, 1997.
[5] D. Dumitrescu, B. Lazzerini, L. Jain and A. Dumitrescu, *Evolutionary Computing*, CRC Press, Boca Raton, 2000.
[6] L. Duta, F. G. Filip, J. M. Henrioud and C. Popescu, Disassembly line scheduling with genetic algorithms, *International Journal of Computers, Communications and Control*, vol.3, no.3, pp.270-280, 2008.
[7] I. Dzitac and B. E. Barbat, Artificial Intelligence + Distributed Systems = Agents, *International Journal of Computers Communications and Control*, vol.4, no.1, pp.17-26, 2009.
[8] L. Eder, *Managing Healthcare Information Systems with Web-Enabled Technologies*, IGI Global, 2000.
[9] S. Fatima, *An Adaptive Organizational Policy for Multi-Agent Systems*, Ph.D. Thesis, University of Hyderabad, 1999.
[10] S. Fatima and G. Uma, An adaptive organizational policy for multi agent systems, *Proc. of the 3rd International Conference on Multi-Agent Systems*, Paris, France, pp.120-127, 1998.
[11] S. Fatima and M. Wooldridge, Adaptive task and resource allocation in multi-agent systems, *Proc. of the 5th International Conference on Autonomous Agents*, Canada, pp.537-544, 2001.
[12] J. Ferber, *Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence*, Addison Wesley, 1999.
[13] D. B. Fogel, *Evolutionary Computation, Toward a New Philosophy of Machine Intelligence*, IEEE Press, New York, 2000.
[14] L. S. Gottfredson, Foreword to intelligence and social policy, *Intelligence*, vol.24, no.1, pp.1-12, 1997.
[15] F. Guichard and J. Ayel, Logical reorganization of DAI systems, *Lecture Notes in Artificial Intelligence*, vol.890, pp.118-128, 1995.
[16] B. Hayes-Roth, An architecture for adaptive intelligent systems, *Artificial Intelligence*, vol.72, no.1-2, pp.329-365, 1995.
[17] B. Hayes-Roth, M. Hewett, R. Washington, R. Hewett and A. Seiver, Distributing intelligence within an individual, *Technical Report*, Distributed Artificial Intelligence, CA, USA, 1989.
[18] J. Huang, N. R. Jennings and J. Fox, An agent-based approach to health care management, *International Journal of Applied Artificial Intelligence*, vol.9, no.4, pp.401-420, 1995.
[19] B. Iantovics, C. Chira and D. Dumitrescu, *Principles of the Intelligent Agents*, Casa Cartii de Stiinta Press, Cluj-Napoca, 2007.
[20] B. Iantovics, *New Paradigms of the Evolutionary Computation, Association with the Intelligent Agents*, Ph.D. Thesis, Babes-Bolyai University, 2004.
[21] B. Iantovics, *Evolutionary Mobile Agents, International Conference European Integration between Tradition and Modernity*, Petru Maior University Press, 2005.
[22] B. Iantovics, Problem solving using evolutionary mobile agents, *Proc. of the 9th National Conference of the Romanian Mathematical Society*, Lugoj, Timisoara, pp.408-420, 2005.
[23] B. Iantovics, Evolutionary reorganization of the centralized multiagent systems, *Proc. of the 2nd International Conference on Economics, Law and Management*, Tg. Mures, pp.139-153, 2006.
[24] B. Iantovics, *Evolutionary Learning Techniques*, Scientific Bulletin of the Petru Maior University, Tg. Mures, XV-XVI, 2003.
[25] B. Iantovics, Agent-based medical diagnosis systems, *Computing and Informatics*, vol.27, no.4, pp.593-625, 2008.
[26] B. Iantovics, Cooperative medical diagnoses elaboration by physicians and artificial agents, in *Understanding Complex Systems*, 2009.
[27] I. F. Imam, Intelligent adaptive agents, *Technical Report WS-96-04*, 1996.
[28] T. Ishida, L. Gasser and M. Yokoo, Organization self design of production systems, *IEEE Transactions on Knowledge and Data Engineering*, vol.4, no.2, pp.123-134, 1992.
[29] S. Kirn, Ubiquitous healthcare: The OnkoNet mobile agents architecture, *Proc. of the Conference on Objects, Components, Architectures, Services, and Applications for a Networked World , LNCS*, vol.2591, pp.265-277, 2003.
[30] G. Lanzola, S. Falasconi and M. Stefanelli, Cooperative software agents for patient management, *Proc. of the AIME Conference*, pp.173-184, 1995.
[31] G. Lanzola, S. Falasconi and M. Stefanelli, Cooperating agents implementing distributed patient management, *Proc. of the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Berlin, Springer-Verlag, vol.1038, pp.218-232, 1996.

[32] M. Maruteri and V. Bacarea, Comparing groups for statistical differences: How to choose the right statistical test, *Biochemia Medica*, vol.20, no.1, pp.15-32, 2010.

[33] M. Maruteri, B. Crainicu and A. Schiopu, RoBioCluster – An open source platform for HPC (High Performance Computing)/Linux Clusters in the biomedical field, *Proc. of the European Federation for Medical Informatics Special Topic Conference & ROMEDINF*, pp.174-177, 2006.

[34] M. Maruteri, B. Crainicu and A. Schiopu, ROSLIMS Live CD – All-in-one cross-platform solution for running biomedical software, *Proc. of the European Federation for Medical Informatics Special Topic Conference & ROMEDINF*, pp.178-181, 2006.

[35] S. Negulescu, C. Zamfirescu and B. Barbat, User-driven heuristics for nondeterministic problems, *Studies in Informatics and Control*, vol.15, no.3, pp.289-296, 2006.

[36] U. Neisser, G. Boodoo, T. J. Bouchard, A. W. Boykin, N. Brody, S. J. Ceci, D. F. Halpern, J. C. Loehlin et al., Intelligence: Knowns and unknowns, *American Psychologist*, vol.51, no.77, 1996.

[37] U. Neisser, G. Boodoo, T. J. Bouchard Jr, A. W. Boykin, N. Brody, S. J. Ceci, D. F. Halpern, J. C. Loehlin, R. Perloff and R. J. Sternberg, Intelligence: Knowns and unknowns, *Annual Progress in Child Psychiatry and Child Development*, 1997.

[38] F. B. Pereira and E. Costa, The influence of learning in the behavior of information retrieval adaptive agents, *Proc. of ACM Symposium on Applied Computing*, New York, pp.452-457, 2000.

[39] F. B. Pereira and E. Costa, How adaptive agents learn to deal with incomplete queries in distributed information environments, *Proc. of the Congress on Evolutionary Computation, Technologies*, vol.1, no.14, pp.1329-1336, 2000.

[40] R. Perloff, R. J. Sternberg and S. Urbina, Intelligence: Knowns and unknowns, *American Psychologist*, vol.51, no.2, pp.77-101, 1996.

[41] A. Trewavas, Green plants as intelligent organisms, *Trends in Plant Science*, vol.10, no.9, pp.413-419, 2005.

[42] A. Trewavas, Mindless mastery, *Nature*, vol.415, no.6874, pp.841, 2002.

[43] A. Ulieru and M. Grabelkovsky, Telehealth approach for glaucoma progression monitoring, *Information Theories and Applications*, vol.10, pp.326-329, 2005.

[44] R. Unland, A holonic multi-agent system for robust, flexible, and reliable medical diagnosis, *Lecture Notes in Artificial Intelligence*, vol.2889, no.2003, pp.1017-1030, 2003.

[45] G. Weiss, *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press Cambridge, Massachusetts London, England, 2000.

[46] D. Yergens, J. Hiner, J. Denzinger and T. Noseworthy, Multiagent simulation system for rapidly developing infectious disease models in developing countries, *Proc. of the 2nd International Workshop on Multi-Agent Systems for Medicine and Computational Biology*, Hakodate, Japan, pp.104-116, 2006.

[47] C. B. Zamfirescu and F. G. Filip, Swarming models for facilitating collaborative decisions, *International Journal of Computers Communications and Control*, vol.5, no.1, pp.125-137, 2010.

Received: 20 May 2018 | Revised: 11 October 2018 | Accepted: 22 October 2018

DOI: 10.1002/widm.1294

WILEY WIREs DATA MINING AND KNOWLEDGE DISCOVERY

ADVANCED REVIEW

# Survey on establishing the optimal number of factors in exploratory factor analysis applied to data mining

Laszlo Barna Iantovics[1] | Corina Rotar[2] | Florica Morar[3]

[1]Informatics Department, University of Medicine, Pharmacy, Sciences and Technology of Targu Mures, Targu Mures, Romania

[2]Computer Science Department, "1 Decembrie 1918" University, Alba Iulia, Romania

[3]Industrial Engineering and Management Department, University of Medicine, Pharmacy, Sciences and Technology of Targu Mures, Targu Mures, Romania

**Correspondence**
Laszlo Barna Iantovics, Informatics Department, University of Medicine, Pharmacy, Sciences and Technology of Targu Mures, Targu Mures, Romania 0000-0001-6254-9291.
Email: ibarna@science.upm.ro

**Funding information**
Funded by the CHIST-ERA programme supported by the Future and Emerging Technologies (FET) programme of the European Union through the ERA-NET funding scheme under the grant agreements, title SOON - Social Network of Machines

In many types of researches and studies including those performed by the sciences of agriculture and plant sciences, large quantities of data are frequently obtained that must be analyzed using different data mining techniques. Sometimes data mining involves the application of different methods of statistical data analysis. Exploratory Factor Analysis (EFA) is frequently used as a technique for data reduction and structure detection in data mining. In our survey, we study the EFA applied to data mining, focusing on the problem of establishing of the optimal number of factors to be retained. The number of factors to retain is the most important decision to take after the factor extraction in EFA. Many researchers discussed the criteria for choosing the optimal number of factors. Mistakes in factor extraction may consist in extracting too few or too many factors. An inappropriate number of factors may lead to erroneous conclusions. A comprehensive review of the state-of-the-art related to this subject was made. The main focus was on the most frequently applied factor selection methods, namely Kaiser Criterion, Cattell's Scree test, and Monte Carlo Parallel Analysis. We have highligthed the importance of the analysis in some research, based on the research specificity, of the total cumulative variance explained by the selected optimal number of extracted factors. It is necessary that the extracted factors explain at least a minimum threshold of cumulative variance. *ExtrOptFact* algorithm presents the steps that must be performed in EFA for the selection of the optimal number of factors. For validation purposes, a case study was presented, performed on data obtained in an experimental study that we made on *Brassica napus* plant. Applying the ExtrOptFact algorithm for Principal Component Analysis can be decided on the selection of three components that were called *Qualitative*, *Generative*, and *Vegetative*, which explained 92% of the total cumulative variance.

This article is categorized under:
Algorithmic Development > Statistics
Algorithmic Development > Biological Data Mining
Algorithmic Development > Structure Discovery

**KEYWORDS**

data reduction, exploratory factor analysis (EFA), establishing the number of extracted factors in EFA, researches performed on complex biological systems, structure detection, statistical methods in data mining

## 1 | INTRODUCTION

Data mining techniques, like clustering, classification and others, are frequently used for analyzing large quantities of data in many domains like agricultural sciences (Chinchuluun, Lee, Bhorania, & Pardalos, 2009; He, Ai, Jing, & Liu, 2016; Majumdar, Naraseeyappa, & Ankalaki, 2017; Mucherino, Papajorgji, & Pardalos, 2009) and plant sciences (Popescu, Noutsos, & Popescu, 2016; Soltis, Nelson, & James, 2018). Frequently data mining tasks include different statistical approaches to analyze data. The books of Kantardzic (2002, 2011) make a comprehensive review of the state-of-the-art techniques and methodologies applied for analyzing very large quantities of data in high-dimensional data spaces, in order to obtain new information that can be used in decision-making. Among others, different statistical approaches are analyzed, like Predictive Regression, Analysis of Variance, Logistic Regression and some others applied in data mining. Data mining it is an approach that could be appropriate even for knowledge discovery tasks. The book of Cios, Pedrycz, Swiniarski, and Kurgan (2007) covers a comprehensive analysis of data mining regarded as a knowledge discovery approach.

In Su and Tsai (2011) and Rousseeuw and Hubert (2011) the problem of statistical outlier detection in the context of the data mining is analyzed. Sometimes outlier values, which could be extremely low or extremely high, may influence the obtained results and conclusions formulated based on those results. Based on this fact, outliers sometimes must be identified and deleted. In order to give just a single simple motivation, it can be mentioned that an outlier in a data set can influence in a high degree the value of the calculated average of those data set. In Arik, Iantovics, and Szilagyi (2017) a method called *OutIntSys* is proposed for the detection of systems with outlier intelligence (extremely low or extremely high) from a set of considered intelligent systems (ISs) specialized in solving a specific set of difficult problems.

In Bock (2002) some basic issues and approaches related to the task of classification are introduced. To this task, the author gives three basic in-depth interpretations: discrimination, clustering, and classification. The author then presents the essential steps of discrimination and clustering methods, outlining the necessity of preprocessing of data. In the performed study, there were established links to different statistical approaches such as prediction and conceptual learning methods that include among others the decision trees.

ISs are appropriate for the solving of many hard computational problems that involves data mining (Iakovidis, Maroulis, & Karkanis, 2006; Iakovidis & Smailis, 2012). There is no universal view on what machine intelligence is and what intelligence metric should measure. The problem of measuring machine intelligence is important based on the fact that the differentiation in intelligence between ISs allows the choosing of the system with the highest intelligence able to solve difficult problems (Iantovics, Emmert-Streib, & Arik, 2017; Iantovics, Rotar, & Niazi, 2018). Measuring machine intelligence frequently must include some data mining methods. In (Iantovics, Dehmer, & Emmert-Streib, 2018) a novel intelligence metric called *MetrIntSimil* is proposed. *MetrIntSimil* is based on a complex analysis of some experimental intelligence evaluation data in order to simultaneously measure the intelligence of a large number of ISs, compare their intelligence and, based on the obtained intelligence measure, classify them in intelligence classes. Systems classified in the same class are able to solve difficult problems with the same intelligence level.

Factor analysis (FA) (Prather et al., 1997; Tufféry, 2011; Wedel & Shi, 2010) is a group of methods that can be used for data reduction and structure detection in the domain of data mining and knowledge discovery. FA has a large variety of applications such as successful job search for engineering college graduates (Kim, Sim, Seo, & Son, 2016) just to mention one of the applications. In order to argue the popularity of FA, the paper (Wedel & Shi, 2010) analyzes its application in marketing research, in perceptual mapping and as a latent structures detection method based on subjective judgments. Exploratory Factor Analysis (EFA) is a type of FA, which is a statistical multivariate technique used to describe variance among a set of observed variables/items. In EFA, the variables are correlated with different statistically significant strength of correlation. This could lead to a potentially lower number of unobserved factors. The unobserved factors can be modeled as a linear combination of the observed variables, and an additional error. The information identified about the interdependencies between observed variables can be used with two purposes: to reduce the set of variables or to classify them.

Many studies (Brown, 2009; Choi, Taylor, & Tibshirani, 2017; Costello & Osborne, 2005; Hayton, Allen, & Scarpello, 2004; Preacher, Zhang, Kim, & Mels, 2013; Song & Belin, 2008) discussed methods for choosing the optimal number of factors. Identification of an innapropriate number of factors may sometimes lead to erroneous conclusions. Some of the criteria elaborated in time are appropriate in many situations and based on this reason they are very frequently applied. In this paper, a comprehensive review of the state-of-the-art related to choosing the optimal number of factors was made. We bring forward an algorithm called *ExtrOptFact*, for the selection of the optimal number of extracted factors. Concretely, *ExtrOptFact* presents the main steps of data analysis that should be performed in case of EFA. Related to the subject of establishing the number of factors, it proposes the application of the three most frequently utilized factor selection methods presented in the literature named, Kaiser Criterion, Cattell's Scree test, and Monte Carlo Parallel Analysis. In the decision regarding the determination of the optimal number of extracted factors it indicates the analysis of the total cumulative variance explained by the

extracted factors. This optimal number of extracted factors is sometimes suggested to have at least a minimal threshold value that must be set based on the so-called research specificity established by the human specialist. In order to outline this affirmation a specific experimental setup was performed. Based on our consideration in particular fields, like research in plants sciences, plants are considered complex biological systems; an important aspect concerns the total cumulative variance explained. The algorithm was applied on data collected as the outcome of experiments realized by us on autumn *B. napus* plant. Both Principal Component Analysis (PCA) and Principal Factor Analysis (PFA) were performed, but the presented experimental study mainly presents the PCA.

The paper is organized as follows: Section 2 presents an overview of EFA, then presents its applications in Subsection 2.1; the Subsection 2.2 is dedicated to a survey on the methods for the selection of optimal number of factors in EFA; Section 3 presents the algorithm applied for the selection of optimal number of factors, including an experimental study for the validation of the algorithm; discussions follow in Section 4; Section 5 presents the conclusions of the paper.

## 2 | OVERVIEW OF EFA APPLIED IN DATA MINING

The term factor analysis was first introduced by Thurstone (Thurstone, 1931). FA includes Confirmatory Factor Analysis (CFA) and EFA. CFA assumes that there is a firm idea about the number of encountered factors, and about which variables most likely will load onto each factor. The purpose of an EFA is to explore the relationships among the variables. It does not have an a priori fixed number of factors. It is based on the supposition that the researcher has a general idea about the intended finding, but does not have yet settled a specific hypothesis. The study presented in this paper is focused only on EFA.

EFA has two principal directions: PCA and PFA. The two main application areas of exploratory factor analytic techniques are: (a) to reduce the number of variables (data reduction) and (b) to detect structure in the relationships between variables. The main purpose of structure detection consists of variables classification. In PCA it is assumed that all the variance in an item/variable should be used in the statistical analysis. However, PCA is a method for data reduction. PFA takes into consideration the variance in an item that it has in common with the other items. However, PFA is used to detect structure. Generally speaking, PCA and PFA methods usually yield similar results related to the extracted numbers of factors (called components in case of PCA).

### 2.1 | Applications of EFA in data mining

This section references some studies and researches that prove the large variety of real-life applications of EFA (Prather et al., 1997; Wedel & Shi, 2010) in some data mining techniques that could include data reduction, structure detection, and knowledge discovery. Presented applications include: agricultural sciences, plant sciences, environmental sciences, and health sciences.

A developed self-assessment instrument called Hypomania Checklist-32 (HCL-32), presented in Angst et al. (2005), is appropriate to identify bipolarity as well as unipolar depression in the human population. In An, Hong, and Kim (2011) the factor structure of the Korean version of the HCL-32 for mood disorder patients is assessed. The study was performed by using both EFA and CFA.

The clinical databases that accumulate large quantities of data about the patients' medical condition are analyzed in the paper (Prather et al., 1997). New medical knowledge can be obtained by discovering relationships and patterns within these data. In the performed study, some techniques of data mining and knowledge discovery in databases were used in order to search for relationships in a large clinical database. In the study, there were used data detained for 3.902 obstetrical patients which were evaluated for factors potentially contributing to preterm birth using EFA. As result, three factors were identified for further exploration. The processes involved in mining a clinical database including data warehousing, and different data analysis are described.

Many remote sensing projects that are oriented toward agriculture, like the one described in Bell and Baranoski (2004), require ground-based measurements of plant reflectance and transmittance. Bell and Baranoski (2004) examine the application of PCA in the storage and reconstruction of plant spectral data. A novel method called by the researchers piecewise principal components analysis (PPCA) is proposed. PPCA takes into account the biological factors that affect the interaction of solar radiation with plants. The reconstructions were performed at a root-mean-square error lower than 1%. The authors concluded that PCA can reduce the dimensionality of plant spectral databases from the visible to the infrared regions of the light spectrum. The PPCA approach can further maximize the accuracy/cost ratio of the storage and reconstruction of plant spectral reflectance and transmittance data.

Gouveia, Parreira, and Martins (2005) present an approach based on factor analysis for identifying the pathogenesis of each autonomic manifestation in a cluster headache (CH). The authors analyzed the type of autonomic symptoms reported by

157 CH patients. They identified three principal components, three different mechanisms underlying autonomic manifestations in CH. The components were called: parasympathetic activation, sympathetic defect (miosis and ptosis), and parasympathetic-mediated effect (nasal congestion, eyelid oedema, and forehead sweating).

Shreck, Getz, and Feenstra (2006) analyze the research question, does "certified organic agriculture" encompass a commitment to "sustainability" that prioritizes social goals? It analyzes the relationship between social sustainability and organic agriculture by drawing attention to issues affecting farmworkers. It presents a case study that included a survey of organic farmers in California, with the topic about the possibility of incorporation of social standards into organic certification criteria. The researchers outline that lukewarm support for social certification within organic agriculture exists among certified organic farmers in California. Organic agriculture necessarily fosters social and economic sustainability for most of the farmers involved. Interviews realized with farmers demonstrate that there are individuals whose practices are atypical. Under some circumstances, an organic production system can be socially, environmentally, and economically sustainable.

Treiblmaier and Pinterits (2010) present a study that includes an EFA to generate and merge attributes of websites. As a result different user groups are identified and classified based on their preferences.

Ji, Chen, Niu, Shang, and Dai (2011) present a novel method for transfer learning in a multiview correspondence perspective. The proposal is called Multiview Principal Component Analysis (MVPCA) approach. The presented experimental evaluation proves that MVPCA can significantly reduce the cross-domain prediction error of a baseline nontransfer method. MVPCA can further improve the performance of the state-of-the-art single-view method.

Some farmers use a lot of pesticides, with a high risk of pesticide poisoning in their cultivation process. This usually is characteristic to those who aim to sell their products to the market. Wichai and Kessomboon (2015) study the hidden factors associated with the farmer's behaviors by using pesticides safely. The research has identified eight factors such as "Economic Reasons"; "Knowledge of Toxicity"; "Social reasons"; "Personal Protection"; "Confidence in residue-free cultivation"; "Eagerness to learn"; and "Risk Perception and Practices".

Fatema, Maznah, and Isa (2015) analyze the spatial variations of the water quality parameters of Merbok estuary. In the performed research for the interpretation of the research results, there were used multivariate statistical techniques, PCA, and cluster analysis. Three parameters were monitored at six sampling stations along the studied river stretch. FA was used for the parameters of the surface and bottom water quality, yielding in the identification of four factors. The four factors explained 68.90% of the total variance of collected datasets.

Ro and Ha (2019) present a study intended to explore the consumers' expectations for novel autonomous cars in Korea. The experimental data was collected from 1.506 potential autonomous car users. For the experimental data analysis, an exploratory and a confirmatory factor analysis was applied. Seven consumer expectations were identified; ethics, licensing, convenience, safety, and cost were found to have a direct effect on attitude, whereas safety, convenience, and cost were found to have a direct effect on intention.

### 2.2 | Survey on the optimal number of factors selection in EFA

In this section, there are presented some representative methods for establishment of the number of extracted factors in EFA. Also, there are mentioned studies that present discussions on the opting between different methods in selecting the optimal number of factors. In EFA, the first extracted factor retains most of the variance, later extracted factors retain less variance. In the scientific literature, there is no unanimously accepted approach for choosing the latest factor with the smallest eigenvalue to be selected.

The problem of choosing the optimal number of factors based on different criteria is discussed in many researchers (Brown, 2009; Choi et al., 2017; Costello & Osborne, 2005; Hayton et al., 2004; Preacher et al., 2013; Song & Belin, 2008). Mistakes at this stage may consist in extracting too many or too few factors. An inappropriate number of extracted factors may have as result the obtaining of erroneous conclusions. Hayton et al. (2004) state three reasons why the decision related to the establishment of the number of extracted factors is so important regarding the less-important aspect of selecting an extraction method or the factor rotation method. Costello and Osborne (2005) presented suggestions for choosing a Factor Extraction Method and establishment of the number of factors retained. The main goal of the study was to collect information that will allow researchers to understand the various choices available through popular software packages, and to make decisions about "best practices" in EFA. In particular, the study provides practical information on making decisions regarding factor extraction, factor rotation, the number of factors to interpret, and sample size. Zwick and Velicer (1986) analyzed the relative robustness of EFA regarding the selection of the extraction method and the factor rotation method.

In Song and Belin (2008) a performed study on the subject of establishing the number of extracted factors in case of missing data is presented. It is discussed how to apply model selection techniques using Akaike's Information Criterion and Bayesian Information Criterion to choose the optimal number of factors when some values are missing. In order to validate the proposal, there is presented a simulation study. It is shown how to select the optimal number of factors for simulated data.

The problem of establishment the optimal number of components in PCA is studied in Choi et al. (2017). With this purpose there are proposed distribution-based methods with exact type 1 error controls for hypothesis testing. The confidence intervals for signals in a noisy matrix with finite samples are constructed. The paper represents an extension of a result presented in Taylor, Loftus, and Tibshirani (2016). The proposed methods are compared with some other existing approaches.

The applicability of cross-validation as a solution to select the optimal number of components in PCA is discussed in Josse and Husson (2012). It outlined the computational cost, the main disadvantage of this approach. In a regression analysis the general cross-validation criterion provides appropriate approximations to leave-one-out cross-validation. It is based on the relation between the prediction error and the residual sum of squares that are weighted by elements of a matrix called projection matrix. Such a relation is established in PCA using an original presentation of PCA with a projection matrix. It defined two cross-validation approximation criteria, and some validations were performed based on simulations.

Minka (2001) proposed an approximate Bayesian model selection criterion for determining the optimal number of components to be retained in PCA. Kazianka and Pilz (2009) presented a modified version of the selection criterion presented in the study of Minka (2001). The modified criterion is compared with various other approaches using some specific simulations. The criterion is used to find the optimal number of principal components in hyper-spectral skin cancer images.

The most frequently used methods for the establishment of the number of extracted factors are the Kaiser Criterion (Kaiser, 1960) and the Scree test proposed by Cattell (1966). These methods/criteria are discussed in many studies and researches (Browne, 1968; Cattell & Jaspers, 1967; Hakstian, Rogers, & Cattell, 1982; Linn, 1968; Raîche, Walls, Magis, Riopel, & Blais, 2013; Tucker, Koopman, & Linn, 1969).

Based on a comprehensive study of the scientific literature, Fabrigar, Wegener, MacCallum, and Strahan (1999) concluded that the earliest criterion proposed by Kaiser (1960) is one of the most-known and most frequently utilized in practice. According to this criterion, only the factors that have eigenvalues greater than 1 are retained for interpretation. The motivation of this fact consists in the explanation that an extracted factor with eigenvalue equal with 1 explains the variance equivalent with an item/variable. Some researchers consider even more restrictive criteria, which limit some of the factors with an eigenvalue greater than one. One of the main critics of Kaiser Criterion consists in the fact that sometimes it retains too many factors.

Scree test proposed by Cattell (1966) is also very frequently used. The Cattell's scree test is a nonnumerical solution to determine the number of factors to retain. It consists of the visual exploration of a graphical representation of eigenvalues. The eigenvalues are presented graphically in descending order and linked with a line. The created graph must be visually examined by a human to determine the point at which the last significant drop or break takes place (see Figure 1). One of the main critics of Cattell's Scree test consists of the fact that sometimes it retains too few factors.

Sometimes the application of the both methods, the Kaiser Criterion and the Cattell's Scree test, is recommended in the determination of the number of extracted factors in order to take a more appropriate decision on the number of extracted factors. Figure 1 presents an example of a Scree test applied to a larger set of variables/items.

Raîche et al. (2013) consider that the graphical nature of the scree test does not accurately allow establishing the number of factors to retain. With this purpose, numerical approaches are proposed. The first presented approach deals with the scree part of the eigenvalues plot. The second presented approach focuses on the elbow part of this plot. In order to compare the efficiency of these approaches with some other previously proposed methods, a simulation study is performed.

Parallel analysis (PA) proposed by Horn (1965) is a Monte Carlo simulation technique that aids researchers in determining the number of factors to retain in EFA. Lautenschlager (1989) and Velicer, Eaton, and Fava (2000) present classical studies related to the PA. In the study of Ledesma and Valero-Mora (2007) PA is considered one of the most recommended methods to deal with the number-of-factors-to-retain problem. PA is included in few software packages and for this reason it is less known by researchers than the Kaiser Criterion and the Scree test proposed by Cattell. “Monte Carlo PCA for Parallel

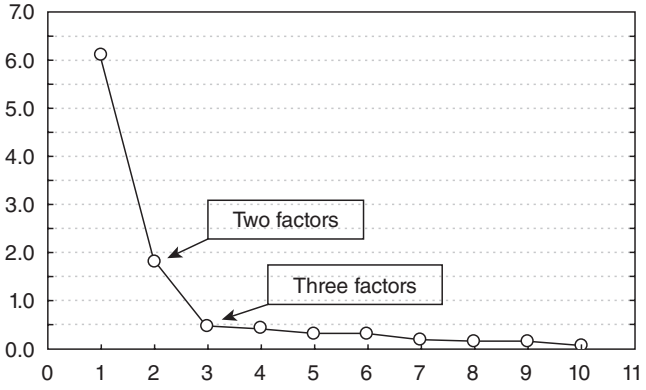



**FIGURE 1** Cattell's scree test. The plot of eigenvalues. X-axis represents the eigenvalue number. Y-axis represents eigenvalues

Analysis" (Watkins, 2000; Watkins, 2006) and "ViSta-PARAN" (Ledesma & Valero-Mora, 2007) are examples of software able to make PA. Watkins (2006) outlines that a main disadvantage of PA is the fact that it makes massive computations that requires large computer resources.

## 3 | EXTROPTFACT ALGORITHM FOR ESTABLISHMENT OF OPTIMAL NUMBER OF EXTRACTED FACTORS IN EFA

### 3.1 | Motivation of experimental researches on *B. napus*

Improving *B. napus* recorded outstanding achievements in recent years, both in terms of increasing the productive potential of an impressive number of cultivars, but especially for increasing the quality of seed. The quality of seeds was deeply modified, as raw material for food oil extraction. It is known that *B. napus* oil can be exploited not just for human consumption but also to produce bio-fuel. The production of bio-fuel is very intensely studied (Hall, Matos, Silvestre, & Martin, 2011). Some recent research have focused on finding solutions to stimulate and intensify the conduct of vegetation stages (autumn and spring). Several growth regulators applied on seeds have been tested, in soil or young plants (in different climatic conditions), in terms of aspect on plants just harvested, but also on the formation of production components, overproduction, content of different fat acids content acids and even the behavior of seeds during storage, and some of them acted in certain concentrations stimulating in all directions (Mondal, Fattah, Latif, & Chondhury, 1996).

### 3.2 | The performed experimental research on *B. napus*

A complex ecosystem is composed of organisms living in a given habitat. An ecosystem is composed of abiotic and biotic components. The subsoil, water, air, climate, and rains are part of the abiotic components. The plants and animals constitute the biotic components. In an ecosystem, there is a set of relationships between the biotic components that inhabit it and the abiotic ones. All these characterize the ecosystem itself. Figure 2 presents a complex ecosystem, which includes a large-size land of *B. napus.* There are some studies related to the biological intelligence of plants. Trewavas (2002, 2005) considered that plants intelligence should be based on principles such as their ability to adjust their morphology, and phenotype accordingly to ensure self-preservation and reproduction. As examples of notable capabilities of the plants that can be considered like some kind of reasoning can be mentioned (Goh, Nam, & Park, 2003; Volkov, Carrell, Baldwin, & Markin, 2009): discriminating negative and positive experiences and learning from past experiences, communication capability, accurate computing their circumstances. Yoneya and Takabayashi (2014) analyze the complex communication between plants from the same or different species. By communicating, plants can form a highly complex system as a whole.

**FIGURE 2** A large-size land of *B. napus* that forms a complex ecosystem

In the experiments performed at *Development Research Center for Cattle Growth* (SCDCB) Mures, growth regulators have been used for the winter *B. napus*, to stimulate the beginning of vegetation (mostly for the stem elongation), because, based on climatic conditions the plants decreased the growing rhythm, and this phase has been extended. Mures County has a continental climate characterized by warm dry summers and relatively cold winters.

Biometric measurements are required in a thorough research as they can correlate with the production (Zamfirescu, 1977). In case of the *B. napus*, many authors (Kumar, Singh, Yadav, & Bikram, 1998; Ozer, Oral, & Dogru, 1999; Zamfirescu, 1977) consider the followings as the most important biometric indicators:

- plant height (height strains);
- the number of branches per plant (including fertile);
- the number capsule on plant;
- the number of seeds on capsule;
- siliques dimensions (characterized by length and diameter); and
- mass of plant seed.

Several authors found a positive correlation between the above components and production of seeds/ha (Kumar et al., 1998; Ozer et al., 1999). These studies find that seed weight correlates most closely with seed production and seed oil percentage.

Based on different studies performed in time we conclude that these production components, although sometimes differ quite a lot from one variety to another, can be modified by the external environmental factors and the influence of various technological factors.

Montvilas (1999) notes that plant density does not influence significantly their size and number of leaves. It very significantly affects the number of the capsule on the plant that decreases with increasing of density with 18–44% from 2 to 4 kg/ha seed. In the case of autumn, varieties interact significantly with density in this regard.

Kutchtova and Vašáak (1998) conclude that with the density, of particular importance is the era of sowing autumn *B. napus*, together with the fertilizers "N," "B," and "Mo" in the formation and reduced generative organs (capsule, seed). In the climatic conditions from Secuieni situated in the Neamt region, Berea (1995) concluded that planting epochs, as well as climatic conditions of the year, can influence not only the length of the growing phases but also the processes that contribute to harvest and which affect the number of capsules on the plant and the average number of seeds per a capsule.

**TABLE 1** Collected biometric indicators data

| Number | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ | Number | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 148.09 | 6.50 | 179.57 | 25.40 | 14.38 | 19 | 146.77 | 6.21 | 154.16 | 25.27 | 11.04 |
| 2 | 151.14 | 6.45 | 135.49 | 24.57 | 10.74 | 20 | 143.38 | 6.29 | 159.20 | 24.85 | 10.30 |
| 3 | 147.01 | 6.53 | 203.47 | 24.46 | 14.72 | 21 | 140.77 | 6.01 | 164.84 | 24.01 | 10.94 |
| 4 | 152.81 | 6.74 | 214.81 | 24.59 | 13.07 | 22 | 143.79 | 5.53 | 133.80 | 23.46 | 9.22 |
| 5 | 146.30 | 5.39 | 112.32 | 23.66 | 7.52 | 23 | 151.78 | 6.29 | 145.37 | 25.29 | 10.44 |
| 6 | 140.37 | 5.85 | 125.57 | 23.74 | 9.17 | 24 | 146.40 | 6.62 | 154.70 | 24.92 | 10.73 |
| 7 | 147.76 | 4.85 | 116.31 | 22.81 | 7.74 | 25 | 143.23 | 5.80 | 147.62 | 25.29 | 13.37 |
| 8 | 145.65 | 6.60 | 164.84 | 23.54 | 10.76 | 26 | 147.82 | 6.82 | 184.59 | 24.97 | 11.82 |
| 9 | 139.59 | 5.15 | 102.32 | 25.59 | 7.59 | 27 | 148.06 | 6.77 | 199.17 | 26.41 | 12.55 |
| 10 | 143.14 | 5.55 | 131.00 | 26.95 | 8.97 | 28 | 143.50 | 6.37 | 181.09 | 25.26 | 12.59 |
| 11 | 143.64 | 4.94 | 129.19 | 26.03 | 9.28 | 29 | 135.87 | 5.83 | 153.76 | 25.03 | 12.03 |
| 12 | 146.30 | 5.55 | 140.13 | 22.89 | 9.46 | 30 | 144.19 | 6.52 | 180.63 | 24.87 | 12.43 |
| 13 | 149.81 | 6.20 | 200.00 | 24.86 | 12.72 | 31 | 146.19 | 5.97 | 136.62 | 24.53 | 10.08 |
| 14 | 155.94 | 6.79 | 196.90 | 25.73 | 11.69 | 32 | 141.47 | 6.12 | 149.44 | 25.56 | 9.82 |
| 15 | 143.31 | 6.61 | 157.66 | 24.69 | 11.08 | 33 | 136.73 | 5.57 | 114.76 | 23.07 | 7.09 |
| 16 | 150.95 | 6.93 | 165.69 | 25.70 | 11.02 | 34 | 135.73 | 5.17 | 132.61 | 22.65 | 8.12 |
| 17 | 141.14 | 5.11 | 158.14 | 24.19 | 12.71 | 35 | 135.81 | 4.73 | 127.39 | 23.23 | 7.39 |
| 18 | 150.54 | 6.51 | 171.40 | 25.14 | 10.91 | 36 | 137.54 | 5.72 | 156.27 | 24.09 | 10.26 |

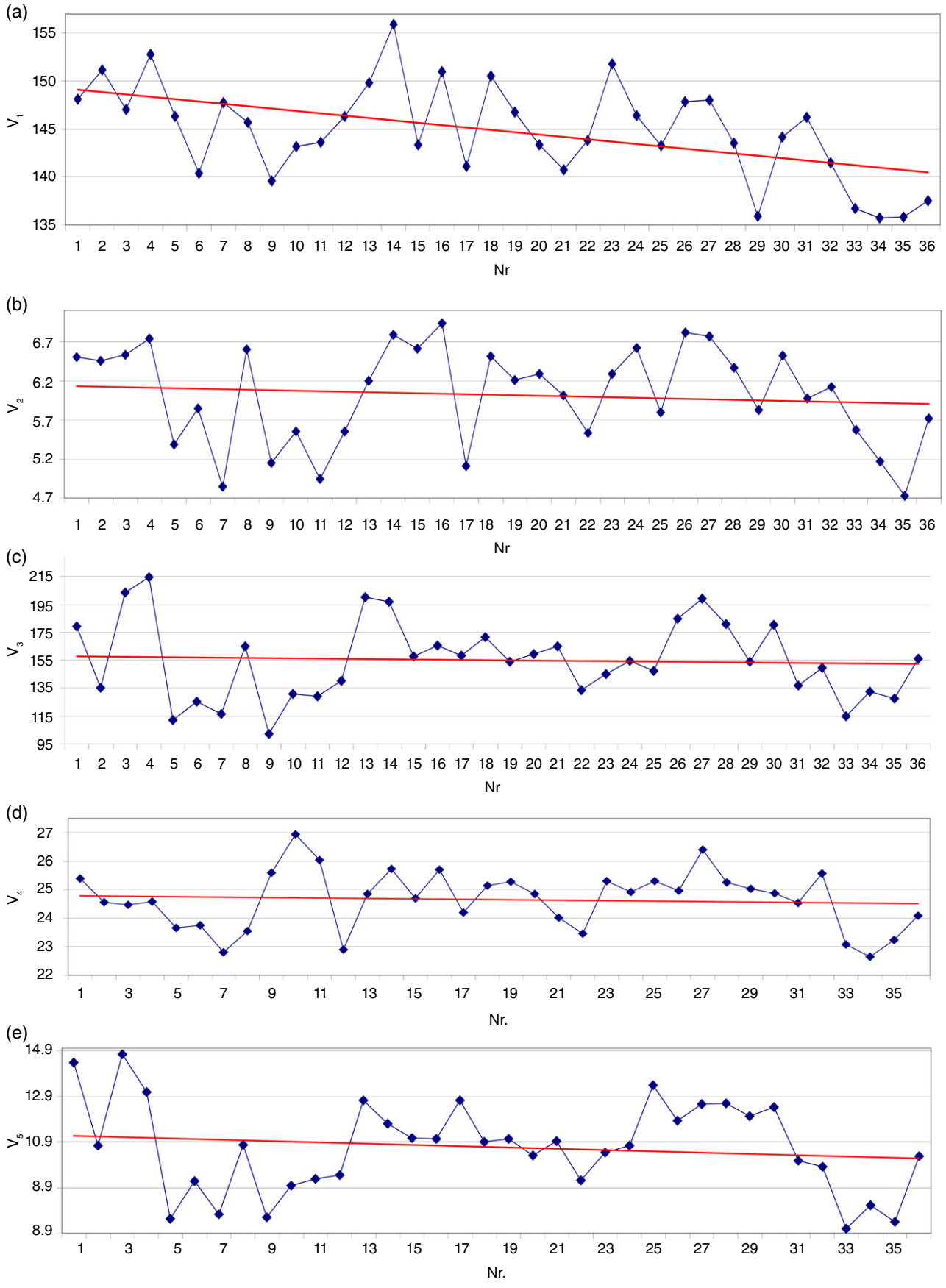



**FIGURE 3** Visual representation of experimental data with the linear trendline included. (a) Graphical $V_1$ variable. (b) Graphical $V_2$ variable. (c) Graphical $V_3$ variable. (d) Graphical $V_4$ variable, and (e) Graphical $V_5$ variable

### 3.3 | Materials and method

To establish the influence of some factors on autumn *B. napus*, biometric indicators were evaluated at *SCDCB* research center from Mures county, more cultivar of autumn *B. napus* (called: Valesca, Digger, and Kardinal), sown at different densities (100; 200; 300 bg/m$^2$.) and administering different chemical fertilizers (called: N0P0K0, N60P60K0, N60P0K0, N90P90K90). Table 1 presents the collected data during the experiments fulfilled on autumn *B. napus* that we used in the experimental study for evaluation purposes. Figures 3 graphically represent the data from Table 1, which allows to humans to make a visual analysis. To make easier the interpretation Figure 3 includes the red plotted linear trendline/linear regression line. A more in-depth description of linear regression and plotting a linear regression line in Kenney and Keeping (1962) is presented. The linear trendline was included just with the purpose to allow a better visual interpretation of data variability. The notations used in Figure 2 and Table 1 are the following: $V_1$ = waist measured in cm; $V_2$ = plant ramification; $V_3$ = plant capsule; $V_4$ = seeds capsule; $V_5$ = weight of plant seeds measured in gr. unit of measure.

For experimental purposes, EFA was realized based on both PCA and PFA on the collected data. In the case of a particular research, the researcher must choose between one of them based on the consideration on which of them is more appropriate to the performed research (is necessary data reduction or structure detection). In Costello and Osborne (2005) the rationale for choosing between PCA and PFA in a performed research that requests EFA is analyzed. In many statistical software packages, PCA is considered as an implicit option.

In the following, just the results obtained during the performed PCA are presented, which has been considered as experimental evaluation study of the algorithm on the used experimental data.

The presentation of PCA was chosen based on the motivation of data reduction. Data reduction could decrease the research costs. PCA assumes that all the variance in an item/variable should be used in the statistical analysis. The presented algorithm called *Optimal Number of Extracted Factors in EFA Applied in Data Mining Algorithm* (*ExtrOptFact*) can be used for the establishment of the optimal number of extracted factors in EFA.

During the statistical analysis, for the data characterization, in the first steps, it was realized as indicated in Marusteri and Bacarea (2010) and Savchenko and Belova (2015) a specific Descriptive Statistics to each of the studied variables. A Descriptive Statistics is useful to describe some of the basic features of the data sets used in a study. Table 2 presents the results of the performed descriptive statistics realized for all the studied variables.

In case of the data of each variable the following notations are used. *N* denotes the sample size, *Min* denotes the smallest value, *Max* denotes the largest value, *Range* is calculated as the difference between *Max* and *Min*, $Range = Max - Min$, *Mean* represents the average of the sample data. [LCI, UCI] denotes the confidence interval of the mean. The most appropriate confidence level of the confidence interval was considered 95%, which is the most usual (Zar, 1984). As confidence levels other values like 90% and 99% can be considered, but they are less frequently used. The establishment of the confidence interval of the mean was necessary based on the fact that a sample was considered, not the whole population. It is impossible to have in most of the situations data of the whole population. Calculated means of different data sets sampled from the whole population have slightly different values. *Standard Deviation (SD)* (Bland & Altman, 1996) is a measure used to quantify the amount of variation of a dataset. *Var*, $Var = SD^2$ denotes the variance, which measures how far a set of numbers are spread out from their average value. *SE* of a parameter is the *SD* of its sampling distribution. SEM denotes the Standard Error of the Mean, $SEM = SD/sqrt(N)$, where *sqrt(N)* denotes the square root, and *Median* represents the median of the experimental evaluation results. Skewness (Joanes & Gill, 1998) is a measure of symmetry (lack of symmetry). *Skewness* indicates symmetric data set if it looks the same to the left and right of the center point. *Skew* denotes the skewness. *Kurtosis* (Joanes & Gill, 1998) can be defined as the measure of whether the data are light-tailed or heavy-tailed relative to a normal distribution. *Kurt* denotes the kurtosis.

In case of the data of each variable the coefficient of variation (*CV*) was calculated, $CV = (SD/Mean) \times 100$, in order to allow the establishment of the data homogeneity–heterogeneity. Let us consider the following parameters $CV_a$, $CV_b$, and $CV_c$. A data set is homogeneous (*hom*), when $CV < CV_a$; relative-heterogeneous (*rel-hom*), when $CV \in [CV_a, CV_b)$; relative

**TABLE 2** Results of the descriptive statistics

| Calculus | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
|---|---|---|---|---|---|
| Mean/SEM | 144.792/0.842 | 6.016/0.104 | 155.023/4.692 | 24.647/0.173 | 10.66/0.33 |
| [LCI, UCI] | [143.1/146.5] | [5.81,6.23] | [145.5, 165.6] | [24.3,25] | [9.9,11.33] |
| Median/*N* | 144.92/36 | 6.16/36 | 154.43/36 | 24.86/36 | 10.75/36 |
| SD/Variance | 5.0509/25.511 | 0.6255/0.391 | 28.1539/792.64 | 1.0358/1.073 | 1.9772/3.909 |
| Min/Max | 135.73/155.94 | 4.73/6.93 | 102.32/214.81 | 22.65/26.95 | 7.09/14.72 |
| CV/homogeneity | 3.5/Hom | 10.4/Hom | 18.2/Rel-hom | 4.2/Hom | 18.6/Rel-hom |
| Range | 20.21 | 2.2 | 112.49 | 4.3 | 7.63 |

heterogeneous (*rel-het*), when $CV \in [CV_b, CV_c)$ and heterogeneous (*het*), when $CV \geq CV_c$. In most of the cases the recommended values for the parameters mentioned above are, $CV_a = 10$, $CV_b = 20$, and $CV_c = 30$.

Data cleaning frequently is necessary as a preprocessing step in data mining. Outlier (extreme, which could be very high or very low) values (statistically different from those others) could significantly influence the obtained results and the formulated conclusions based on those results. Sometimes outlier detection and elimination is necessary during data cleaning.

In the studied variables (each variable consisting in a data sample) the presence of outliers was analyzed using the Grubbs test for outliers' detection (Barnett & Lewis, 1994). The Grubbs test is appropriate if the data normality is expected (a studied data sample is expected to follow an approximately normal distribution). In statistics, if the estimated value is statistically significantly higher or lower than the rest of the values, a two-tailed test (alternative name two-sided test) should be applied. In other situations, a one-tailed test (alternative name one-sided test) should be applied. The one-tailed test is appropriate if the estimated value may depart from the reference value in only one direction. In the performed case study the two-sided version of the test was applied, with the significance level $\alpha Grubbs = 0.05$. The significance level by 0.05 is recommended in most of the cases. The applied Grubbs test for outliers' detection does not indicate the presence of outliers in case of any of the studied variables.

As an alternative to the Grubbs test, sometimes the *Tietjen-Moore* test (Tietjen & Moore, 1972) can be applied. The main limitation of the Tietjen–Moore test consists in the fact that the number of outliers is necessary to be specified exactly at the application of the test. This supposes that this number must be known in advance, which very rarely is possible in practical situations. Tietjen–Moore test similarly with the Grubbs test is based on the assumption of data normality. The Generalized Extreme Studentized Deviate (ESD) test (Rosner, 1983) requires an upper bound on the prospective number of outliers. Its use is recommended when there are one or more outliers but the exact number of outliers is not known in advance.

We consider the following indicator values of *Skew* and *Kurt*. A skewness value, *Skew* with $|Skew| > 1.96$, means that the skewness is extremely deviated from zero. A kurtosis value *Kurt* with $|Kurt| > 1.96$, means that the kurtosis is extremely deviated from zero.

The histogram was first introduced by Karl Pearson (Pearson, 1895b). We recommend the use of histograms as a graphical technique for different kind of visual interpretations, expressive in case of larger data sets, less expressive in case of smaller data sets. The humans and computing systems have different strengths and weakness comparatively with each other. Frequently easier from the human side of view as compared to computing systems, is formulating some conclusions or taking decisions based on visual interpretation of graphically represented data. Among others, a histogram is useful for visual showing the skewness and kurtosis of a studied experimental evaluation data set and an orientative visual examination of data normality.

The most well-known goodness-of-fit tests of normality are Razali and Wah (2011) and Stephens (1974): *One-Sample Kolmogorov–Smirnov test (KS test)*, *Shapiro–Wilk test* (*SW test*), *Lilliefors test* (*Lill test*) (i.e., based on the KS test), and Anderson–Darling test (AD test). KS test is one of the most frequently used. In Razali and Wah (2011) all the above-mentioned tests of normality are studied, and proved that the *SW* test is the most powerful, but it has some disadvantages as well. For testing the data normality (sampling from a Gaussian population), the KS test (Chakravarti, Laha, & Roy, 1967) and the SW test (Razali & Wah, 2011) were applied, with the $\alpha Norm = 0.05$ significance level, results presented in Table 4. The significance level by 0.05 is considered the most appropriate in many studies (Chakravarti et al., 1967; Razali & Wah, 2011; Stephens, 1974). Let us denote with *Pnorm*, the obtained *p*-value of an applied normality test. If $Pnorm > \alpha Norm$, then the normality assumption passed at the $\alpha Norm$ significance level, elsewhere ($Pnorm \leq \alpha Norm$) the normality assumption failed at the $\alpha Norm$ significance level. *Scatterplots* are frequently used for visual examinations (Friendly & Denis, 2005). The Quantile–Quantile plot (QQ Plot) is a scatterplot with some kind of specificity (Wilk & Gnanadesikan, 1968). A QQ Plot is created by plotting two sets of quantiles against one another. *QQ Plot* is recommended for the visual appreciation of normality. If both sets of quantiles came from the same distribution, the points form a line that is roughly straight. The joint use of QQ plot with the *SW test* is suggested for accurate verification of the normality assumption.

The visual analysis of histograms, Figure 4a (for $V_1$ variable), Figure 5a (for $V_2$ variable), Figure 6a (for $V_3$ variable), Figure 7a (for $V_4$ variable), Figure 8a (for $V_5$ variable), indicates the $V_1$, $V_2$, $V_3$, $V_4$ and $V_5$ normality and the fact that have sense to make some further verifications of the normality using statistical goodness-of-fit tests and other representations appropriate to make a visual analysis. Table 3 presents the calculated skewness and kurtosis for $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$. All the absolute value of calculated kurtosis and skewness for $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$ were lower than 1.96 ($|Kurt| < 1.96$, $|Skew| < 1.96$) that indicate that there is no extreme deviation from 0.

In case of the *SW* test as a secondary analysis there were visually analyzed the QQ Plots, Figure 4b (QQ Plot of $V_1$), Figure 5b (QQ Plot of $V_2$), Figure 6b (QQ Plot of $V_3$), Figure 7b (QQ Plot of $V_4$), and Figure 8b (QQ Plot of $V_5$). All the QQ Plots visually indicate the passing of normality assumption. All the performed numerical (Table 4) and visual analyses of $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$ normality indicate the passing of the normality assumption at the $\alpha Norm = 0.05$ significance level. As an observation it can be noticed that the *Pnorm* provided by the *SW* test is relatively close to 0.05. Based on the fact that *SW test* is the most powerful normality test it can be considered that the normality assumption passed. This is enforced by the interpretation

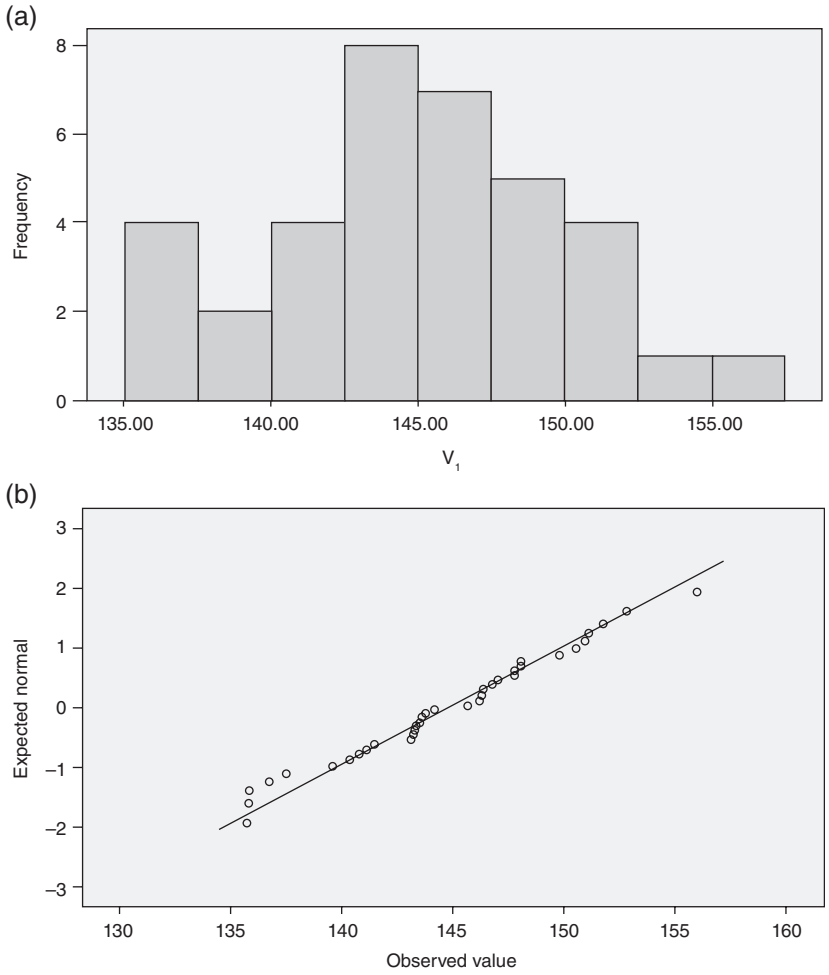



**FIGURE 4** Visual analysis of $V_1$ normality. (a) Histogram of $V_1$; (b) QQ plot of $V_1$

of the QQ plot associated to $V_2$ variable that allows the formulation of the same conclusion related to the data normality ($V_2$ passes the normality assumption).

Important assumptions that should be verified for the applicability of *EFA* are the Kaiser–Meyer–Olkin Measure of Sampling Adequacy (*KMO*) and Bartlett's Test of Sphericity (BTS). *Kmo* denotes *the Kaiser-Meyer-Olkin Measure of Sampling Adequacy* that the test provides as result, $Kmo \in [0, 1]$. The assumption of *KMO* test passed if $Kmo \geq 0.6$. The assumption of *Bartlett's Test of Sphericity* is passed if the obtained *BtsSig* as a test result is $BtsSig < 0.05$. Table 5 presents the results of *KMO* and *BTS* tests applied on the studied experimental data.

As a next step, the Pearson Product Moment Correlation (Pearson, 1895; Stigler, 1989) was applied, developed by Karl Pearson based on the idea of Francis Galton. Table 6 presents the obtained correlation matrix, between all the measured variables $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$. In order to maintain the generality, the method of mean substitution of missing data (in case of our data there were no missing values) was applied. As it is well-known, a correlation coefficient denoted $r$ indicates the correlation strength between two variables, it should take values in the interval $[-1, 1]$ ($r \in [-1, 1]$).

It was applied a posttest, which proved that all the correlations are significant at the established $\alpha Cor = 0.05$ significance level. The statistical significance of a correlation coefficient that is statistically different from zero was established by comparing the obtained *Pcor* (*p-value* of the statistical significance test) with the $\alpha Cor$, and the decision is based on the $Pcor < \alpha Cor$. A requirement for the *Determinant of Correlation Matrix (DCM),* calculated value denoted *Dcm* is that $Dcm > 0.001$. The obtained *Dcm* of the correlation matrix was 0.054, which satisfy the criteria of $Dcm > 0.001$.

In the following, just the results of factor extraction by performing PCA are presented. For the establishment of the number of extracted components, first, we analyzed the most frequently used factor selection methods: Kaiser Criterion, Scree test proposed by Cattell and PA.

Table 7 presents the eigenvalues of all the possible five components. The fact that the eigenvalue of the first component is greater than the eigenvalue of the second component was expectable. As it is known, the first component extracts most of the variance, following components extract less and less of the total variance. Based on the Kaiser Criterion (Fabrigar et al.,

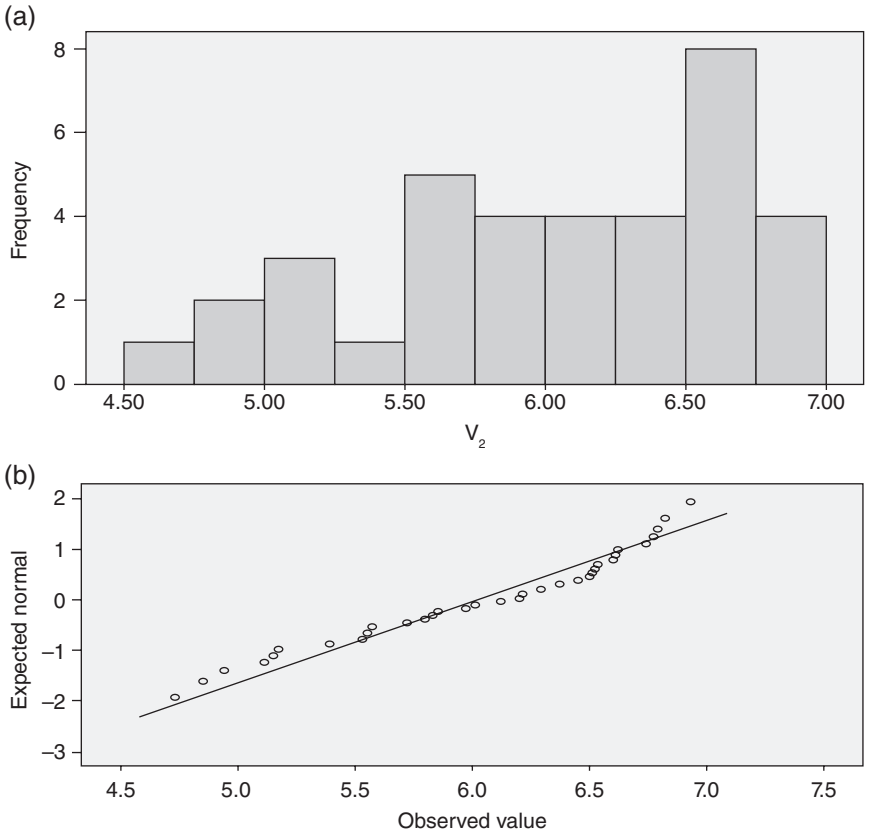



**FIGURE 5** Visual analysis of $V_2$ normality. (a) Histogram of $V_2$; (b) QQ plot of $V_2$

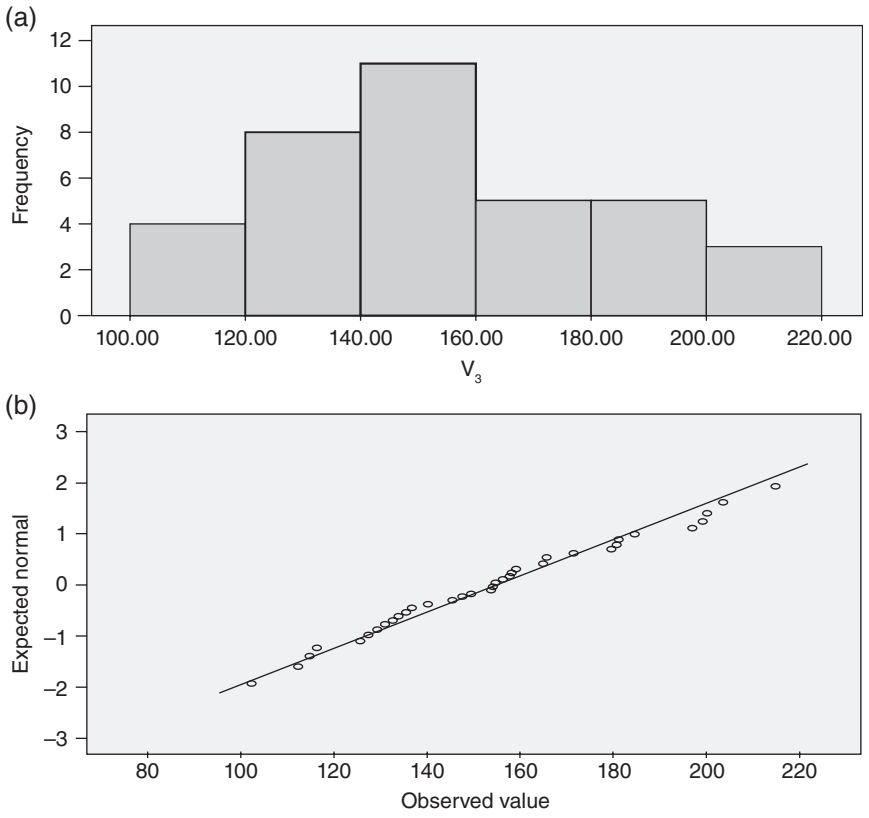



**FIGURE 6** Visual analysis of $V_3$ normality. (a) Histogram of $V_3$; (b) QQ plot of $V_3$

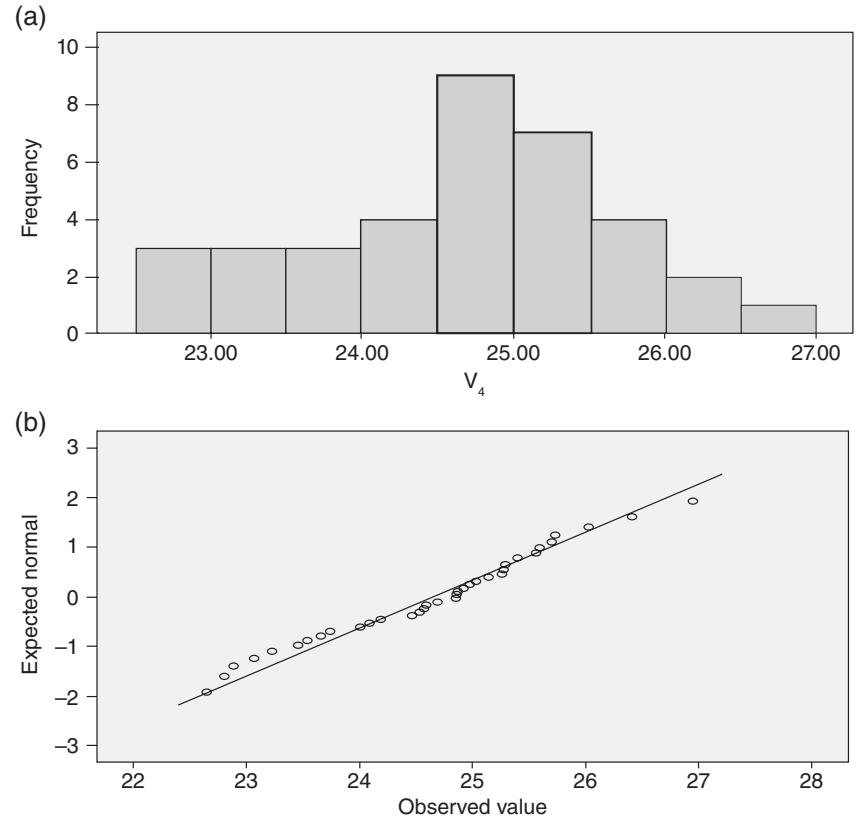



**FIGURE 7** Visual analysis of $V_4$ normality. (a) Histogram of V4; (b) QQ plot of $V_4$

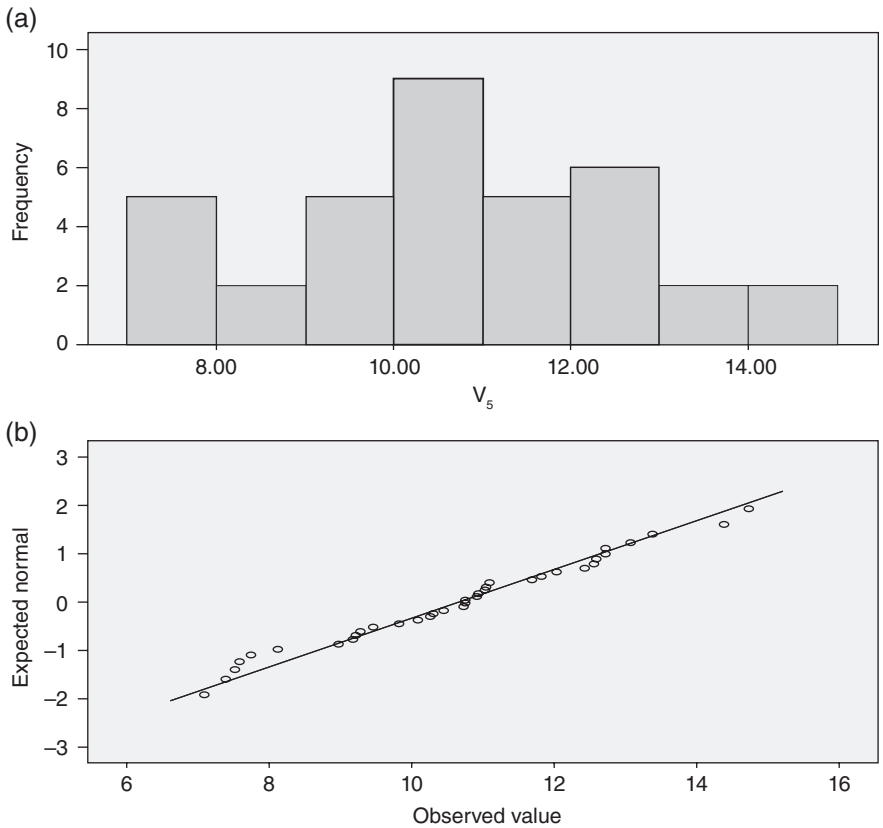



**FIGURE 8** Visual analysis of $V_5$ normality. (a) Histogram of $V_5$; (b) QQ plot of $V_5$

**TABLE 3** Calculated skewness and kurtosis

| Performed calculus | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
|---|---|---|---|---|---|
| *Kurt* | −0.384 | −0.883 | −0.566 | −0.357 | −0.544 |
| *\|Kurt\| > 1.96* | No | No | No | No | No |
| *Skew* | −0.065 | −0.471 | 0.266 | −0.152 | −0.013 |
| *\|Skew\| > 1.96* | No | No | No | No | No |

**TABLE 4** KS and SW tests results for $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$ data, at $\alpha Norm = 0.05$ significance level

| Performed calculus | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
|---|---|---|---|---|---|
| KS test | | | | | |
| *KS stat/Pnorm* | 0.094/>0.1 | 0.117/>0.1 | 0.0799/>0.1 | 0.105/>0.1 | 0.083/>0.1 |
| Normality assumption passed (*Pnorm*>α*Norm*) | Yes | Yes | Yes | Yes | Yes |
| SW test | | | | | |
| *SW stat/Pnorm* | 0.976/0.609 | 0.942/0.057 | 0.978/0.665 | 0.976/0.619 | 0.975/0.577 |
| Normality assumption passed (*Pnorm*>α*Norm*) | Yes | Yes | Yes | Yes | Yes |

**TABLE 5** Results of the *KMO* and *BTS* tests

| | |
|---|---|
| Kaiser-Meyer-Olkin measure of sampling adequacy | |
| *Kmo* | 0.756 |
| *Test assumption passed* (*Kmo* ≥ 0.6) | Yes |
| Bartlett's test of Sphercity | |
| *Approximate Chi-square* | 94.899 |
| *BtsSig* | ≈0 |
| *Test assumption passed* (*BtsSig* < 0.05) | Yes |

**TABLE 6** Correlation matrix

| | $V_1$ | $V_2$ | $V_3$ | $V_4$ | $V_5$ |
|---|---|---|---|---|---|
| $V_1$ | — | 0.62 | 0.51 | 0.35 | 0.42 |
| | — | *Pcor*≈0 | *Pcor*≈0.002 | *Pcor*≈0.04 | *Pcor*≈0.01 |
| $V_2$ | 0.62 | — | 0.75 | 0.43 | 0.66 |
| | *Pcor*≈0 | — | *Pcor*≈0 | *Pcor*≈0.009 | *Pcor*≈0 |
| $V_3$ | 0.51 | 0.75 | — | 0.35 | 0.85 |
| | *Pcor*≈0.002 | *Pcor*≈0 | — | *Pcor*≈0.037 | *Pcor*≈0 |
| $V_4$ | 0.35 | 0.43 | 0.35 | — | 0.42 |
| | *Pcor*≈0.037 | *Pcor*≈0.009 | *Pcor*≈0.037 | — | *Pcor*≈0.01 |
| $V_5$ | 0.42 | 0.66 | 0.85 | 0.42 | — |
| | *Pcor*≈0.01 | *Pcor*≈0 | *Pcor*≈0 | *Pcor*≈0.01 | — |

1999), if we take into account just components with eigenvalue greater or equal with 1, then just a single component can be identified. There are different proposals that limit even the components with eigenvalue greater than 1.

Figure 9 presents the eigenvalues graphically in descending order and linked with a line. This is necessary for making the Cattell Scree test of visual analysis. Based on the visual analysis, checking the point at which the last significant drop or break takes place, it can be concluded the clear choosing of one component, the other two next components contributing in a smaller degree to the value of total cumulative variance explained.

Also, the third often used test by PA was realized. In the case of PA, there were generated random eigenvalues. Table 8 presents the eigenvalues obtained by PA using Monte Carlo method based on 100 replications. *GRE* denotes the obtained eigenvalues, calculated as the average of the 100 samples of generated random eigenvalues. SDE denotes the standard deviation. At the component selection must be selected just the components with an eigenvalue greater than the generated random eigenvalues. As can be noticed the obtained results suggest the choosing of a single component.

**TABLE 7** Eigenvalues obtained by applying the principal components extraction

| Order | Component | Eigenvalue | % Total variance | Cumulative eigenvalue | Cumulative % |
|---|---|---|---|---|---|
| First extracted | $*Fact_1$ | 3.198 | 63.967 | 3.1984 | 63.967 |
| Second extracted | $*Fact_2$ | 0.747 | 14.935 | 3.945 | 78.902 |
| Third extracted | $*Fact_3$ | 0.65 | 13.007 | 4.595 | 91.909 |
| Fourth extracted | $Fact_4$ | 0.281 | 5.619 | 4.876 | 97.528 |
| Fifth extracted | $Fact_5$ | 0.124 | 2.472 | 5 | 100 |

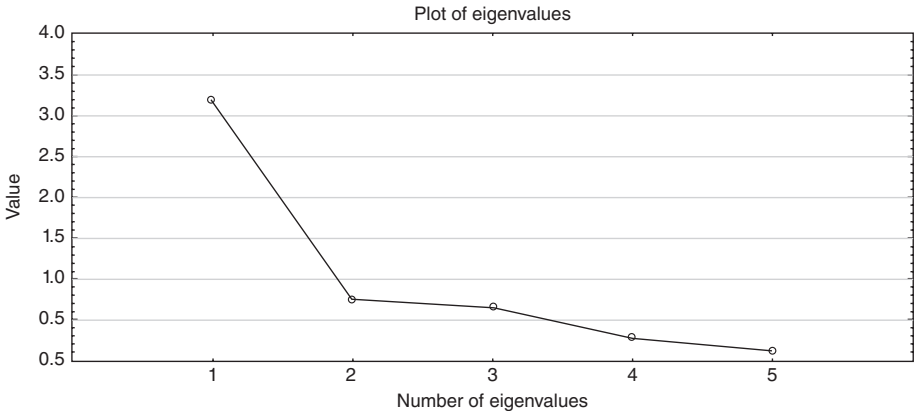



**FIGURE 9** Plot of eigenvalues. Scree test proposed by Cattell

**TABLE 8** Eigenvalues obtained by Monte Carlo parallel analysis

| No. | GRE | SDE | Eigenvalue | Eigenvalue > GRE | Indication of component selection |
|---|---|---|---|---|---|
| 1 | 1.2209 | 0.0559 | *3.198* | Yes | Yes |
| 2 | 1.0958 | 0.0367 | *0.747* | No | No |
| 3 | 1.0024 | 0.0327 | *0.65* | No | No |
| 4 | 0.8951 | 0.0428 | *0.281* | No | No |
| 5 | 0.7858 | 0.0450 | *0.124* | No | No |

In the following, the algorithm called Optimal Number of Extracted Factors in EFA Applied in Data Mining Algorithm (*ExtrOptFact)* is described, for the establishment of the optimal number of factors that must be extracted. The particularities of the approached research, that we will generically call in the following *Research Specificity* (*RS*), were: research on plants that are complex biological life forms, which have a large variability that is specific to biological systems; a limited number of studied variables. RS should be established by a human specialist (HS). HS is a specialist that detains in-depth knowledge (has the necessary expertise) about the performed experimental research, research data and statistical data analysis. Usually, these two types of expertise do not exist in a single human. However, HS is defined as the human (or humans) that detain both expertises. In case of the obtained experimental *B. napus* data, HS denotes the specialist who detains the knowledge and expertise related to the plant data (knowledge related to *B. napus*), and the statistical analysis. This allows the establishment of the required total cumulative variance to be explained by the extracted components. This does not suppose mathematical calculus, it is based on the knowledge about the specific of the data (as result of specialty knowledge/expertise, experience and human intuition) and knowledge about statistical data analysis.

The value of input parameter of the algorithm in our experimental study are the $N$ variables, $|V| = N = 5$; $V = \{V_1, V_2, V_3, V_4, V_5\}$. The output of the algorithm in our experimental study is $M = 3$ which denotes that by applying the algorithm three factors (they are called factors just for illustrating the general idea, practically they are components based on the fact that a PCA is performed) are selected (the first three components). The identified factors are denoted $IDFactors = \{Fact_1, Fact_2, Fact_3\}$; where $Fact_1$, $Fact_2$, and $Fact_3$ are uncorrelated orthogonal factors, they are independent of each other. The selected first three factors are marked in Table 7 with "*". The eigenvalue of the first factor denoted $Fact_1$ is by ≈3.198 that explain ≈63.967% of the total variance. The eigenvalue of the second factor denoted $Fact_2$ is by ≈0.747 that explain ≈14.935% of the total variance. The eigenvalue of the third factor denoted $Fact_3$ is by ≈0.65 that explain ≈13% of the total variance. As can be noticed analyzing the algorithm, there are admitted even factor with eigenvalue lower than 1 in some situations based on the specificity of the performed research.

***ExtrOptFact***

***Optimal Number of Extracted Factors in EFA Applied in Data Mining Algorithm***

**IN:** $V=\{V_1, V_2, \ldots, V_n\}$; //Variables from that should be extracted the factors (or components). If is performed a PCA the correct name is "component". If is performed a PFA the correct name is "factor". Sometimes in order to maintain the generality of explanation is used the name factor.

**OUT:** *M*; //The number of identified factors (or components).

**Step 1:** *Preprocessing and data cleaning. Statistical characterization of* $V_1, V_2, \ldots, V_n$;

@Calculates for each variable the:

*Mean*, *SEM*, [*LCI, UCI*], *Median*, *SD*, *Variance*, *Kurt*, *Skew*, *Min*, *Max*, *CV*, *Range*;

@Verify the normality of each studied variable;

@Search for outliers and if identified, and considered necessary eliminate them;

**Step 2:** *Verification of the applicability of EFA.*

@Calculates *Kmo*; //Verification of the *KMO* test assumption.

@Calculates *BtsSig*; //Verification of the *BTS* test assumption.

@Calculates the *Correlation Matrix*;

@Calculates *Dcm*;

**If** ((*Kmo* $\geq$ 0.6)**and**(*BtsSig* $<$ 0.05)**and**(*Dcm* $>$ 0.001)) **Then**

@There are continued the following steps of the algorithm.

**Else** //The verification of the necessary assumptions for the application of EFA failed.

@The execution of the algorithm is interrupted. **Print** "EFA could not be performed";

**EndIf**

**Step 3:** *Establishment of the necessary type of EFA* (*PCA* or *PFA*).

@Establishment by the *HS* the necessary *EFA* to be applied;

**If** (is necessary a method for data reduction) **Then** *FactAnal:*= "PCA";

**Else** *FactAnal:*= "PFA"; //Is necessary a method for structure identification.

**EndIf**

**Step 4:** *Performing the factor (or component) extraction.*

@Apply the selected *FactAnal* method for extracting the factors (or components).

@Establishes an appropriate *rotation strategy.*

@Rotate the factors (or components) based on the established rotation strategy.

@Obtains the factor (or components) *loadings* and *communalities.*

**Step 5:** *Establishment of the optimal number of factors* (or components) *to be extracted.*

**Step 5.1:** Apply the Kaiser Criterion, Scree test and Parallel Analysis.

@Presents the result of Kaiser Criterion to *HS*;

@Makes the visual representation according to Scree test to be evaluated by *HS*;

@Performs and presents the results of *Parallel Analysis* to *HS*;

@*HS* analyzes the number of factors (or components) suggested to be extracted and based on that formulates some conclusions;

**Step 5.2.** *Analysis of Research Specificity by HS.*

@*HS* analyses the research specificity;

**Step 5.3.** *Establishment of the optimal number of extracted factors.*

@*HS* establishes the total cumulative variance that is considered to be explained based on *RS* and considering the results obtained at the steps **Step 5.1** and **Step 5.2** establishes the optimal number *M* of extracted factors (or components).

***EndExtrOptFact***

The motivation of necessity of factor rotation is based on the fact that unrotated results coming from factor analysis are difficult to interpret. Based on a comprehensive analysis of literature (Cooley & Lohnes, 1971; Harman, 1976; Kim & Mueller, 1978a, 1978b; Lawley & Maxwell, 1971; Lindeman, Merenda, & Gold, 1980; Morrison, 1967; Mulaik, 1972; Stevens, 1986; Wherry, 1984) it can be concluded that frequently used factor rotation strategies are the so-called: varimax raw, varimax normalized, biquartimax raw, biquartimax normalized, quartimax raw, quartimax normalized, equamax raw, and equamax normalized. In the technical note (Osborne, 2015) a comprehensive analysis of the significance of rotation is realized, explaining why it is necessary when performing EFA. The utility and desirability of different rotation methods are discussed in detail.

The correlations between the variables and the factors are called *factor loadings.* Table 9 presents the factor loadings, principal components of the extraction method, having as result three factors extracted. The processing for performing the rotation converged in five iterations. It was chosen the application of the *varimax normalized* orthogonal rotation strategy. Varimax normalized rotation is aimed at maximizing the variances of the squared normalized factor loadings across variables for each

**TABLE 9** Rotated component matrix. Factor loadings

| Variable | $Fact_1$ | $Fact_2$ | $Fact_3$ |
|---|---|---|---|
| $V_1$ | 0.231 | 0.150 | **0.938**[a] |
| $V_2$ | **0.667**[a] | 0.214 | 0.557 |
| $V_3$ | **0.910**[a] | 0.108 | 0.294 |
| $V_4$ | 0.203 | **0.964**[a] | 0.165 |
| $V_5$ | **0.917**[a] | 0.227 | 0.132 |
| Expl.Var | 2.213 | 1.061 | 1.321 |
| Prp.Totl | 0.4426 | 0.2123 | 0.2642 |

[a] Marked loadings are >0.55.

factor. This rotation strategy was proposed by Kaiser (Kaiser, 1958). The decision for this factor rotation was based on the fact that the variables were not highly correlated with each other (Table 7 presents the correlation matrix).

In case of studying the strength of a correlation, for absolute values of *r*, a classification can be realized as follows:

- $|r| \in [0, 0.2)$, indicates a very weak correlation;
- $|r| \in [0.2, 0.4)$, indicates a weak correlation;
- $|r| \in [0.4, 0.6)$, indicates a moderate correlation;
- $|r| \in [0.6, 0.8)$, indicates a strong correlation; and
- $|r| \in [0.8, 1]$, indicates a very strong correlation.

Table 10 presents the same rotated component matrix as Table 9 but with the small coefficients ($\leq 0.3$) suppressed and variables ordered. Table 11 presents the calculated communalities. *Communalities* are the proportion of each variable's variance that can be explained by the factors.

## 4 | DISCUSSION

PCA was performed in the experimental study, and we have studied the problem of optimal number of components to be extracted. We used in some places the name factor in the description from above just to maintain the generality of the description. All the methods/criteria for the establishment of the optimal number of extracted components: Kaiser Criterion, Scree test, and Parallel Analysis suggest the choice of one component. Using the *ExtrOptFact* algorithm the first three components were selected. As can be noticed, the second component has the eigenvalue by 0.747; considered by some researchers not relevant (see the Kaiser Criterion; eigenvalue $0.747 < 1$) and the third component has the eigenvalue by 0.65 ($0.65 < 1$). We call the first component *Qualitative Component*, which explains 63.967% of the total cumulative variance. We call the second component *Generative Component*, which explains 14.935% of the total variance. We call the third component *Vegetative Component*, which explains 13% of the total variance. The names of the components were established using human specialty knowledge about agricultural sciences, plants sciences, and plant biology. The total cumulative variance explained by the first

**TABLE 10** Rotated component matrix, with small coefficients suppressed

| | $Fact_1$ | $Fact_3$ | $Fact_2$ |
|---|---|---|---|
| $V_5$ | **0.917** | | |
| $V_3$ | **0.910** | | |
| $V_2$ | **0.667** | **0.557** | |
| $V_1$ | | **0.938** | |
| $V_4$ | | | **0.964** |

**TABLE 11** Calculated communalities

| Variable | Initial | Extraction |
|---|---|---|
| $V_1$ | 1 | 0.955 |
| $V_2$ | 1 | 0.805 |
| $V_3$ | 1 | 0.927 |
| $V_4$ | 1 | 0.998 |
| $V_5$ | 1 | 0.910 |

three components is by 92%. The obtained result by naming of the three components allows the validation of the result obtained by applying the *ExtrOptFact* algorithm on the studied variables $V_1$, $V_2$, $V_3$, $V_4$, and $V_5$.

As an additional explanation, it can be mentioned that frequently in decisions taken related to number of extracted factors the Kaiser Criterion is considered the first applied, followed by the Cattell Scree test. Figure 1 presents an example of a Cattell Scree test applied to 10 variables performing an EFA. By analyzing the visual representation a decision can be made on the identification of three factors. Kaiser Criterion applied alone suggests two factors (the third factor has an eigenvalue lower that 1). By combining Cattell's Scree test result with the Kaiser Criterion result can be decided on two factors.

The *ExtrOptFact* algorithm includes at a decision point the consideration of Research Specificity that is established by the Human Specialist. This makes the algorithm a hybrid one. Different complex decisions are more accurate if they are based on some kind of hybridization. In the presented algorithm the hybridization is based on data processing (Kaiser Criterion, Parallel Analysis), visual analysis (Cattell Scree test) by humans and human intervention in different points of decision. In the performed experimental study we considered that in our particular research topic approached the 63.97% variance explained is not enough and it would be necessary at least 80–90% explanation of the total cumulative variance. This consideration indicated the necessity of choosing the third component, this yielding as cumulative variance explained by ≈92%. We would like to outline the necessity in a research to have a very good understanding of the research to be performed and the specificity of the obtained research data in case of EFA.

## 5 | CONCLUSIONS

Data mining techniques are frequently applied to analyzing large quantities of data in different domains like the agricultural sciences and plants sciences. Some of the data mining techniques should be combined with statistical data analysis. EFA is one of such frequently applied methods. Data mining techniques that require EFA include PCA and PFA. EFA is important in different types of researches that require data reduction or structure identification (frequently necessary in classification). The number of factors to retain is very important in EFA. The survey presented in this paper is mainly focused on some of the most utilized methods, namely Kaiser Criterion, Cattell Scree test, and Parallel Analysis, for the determination of the optimal number of factors in EFA. Each of them has may raise different critics in the specialized literature, such as the selection of an inappropriate number of factors (too few or too many) or the requirement of large computing time. The combination of these methods frequently allows a good choice of the optimal number of factors. In order to choose the optimal number of factors we consider that sometimes, based on the specificity of the research, the total cumulative variance explained by the extracted factors must be considered, which in some case must necessarily have at least a minimal admitted value.

### ACKNOWLEDGMENTS

This work has been funded by the CHIST-ERA programme supported by the Future and Emerging Technologies (FET) programme of the European Union through the ERA-NET funding scheme under the grant agreements, title SOON - Social Network of Machines.

### CONFLICT OF INTEREST

The authors have declared no conflicts of interest for this article.

### RELATED WIREs ARTICLES

Multivariate methods
Multiple factor analysis: principal component analysis for multitable and multiblock data sets
STATIS and DISTATIS: optimum multitable principal component analysis and three way metric multidimensional scaling
Anomaly detection by robust statistics
Statistical data mining

### REFERENCES

An, D., Hong, K. S., & Kim, J. H. (2011). Exploratory factor analysis and confirmatory factor analysis of the Korean version of hypomania Checklist-32. *Psychiatry Investigation*, *8*(4), 334–339.

Angst, J., Adlofsson, R., Benazzi, F., Gamma, A., Hantouche, E., Meyer, T. D., … Scott, J. (2005). The HCL-32: Towards a self-assessment tool for hypomanic symptoms in outpatients. *Journal of Affective Disorders*, *88*, 217–233.

Arik, S., Iantovics, L. B., & Szilagyi, S. M. (2017). OutIntSys - A novel method for the detection of the most intelligent cooperative multiagent systems. In D. Liu, S. Xie, Y. Li, D. Zhao, & E. S. El-Alfy (Eds.), *Neural information processing. ICONIP 2017 Lecture Notes in Computer Science* (Vol. 10637, pp. 31–40). Cham, Switzerland: Springer.

Barnett, V., & Lewis, T. (1994). *Outliers in statistical data* (3rd ed.). Chichester, UK: Wiley.
Bell, I. E., & Baranoski, G. V. G. (2004). Reducing the dimensionality of plant spectral databases. *IEEE Transactions on Geoscience and Remote Sensing*, *42*(3), 570–576.
Berea, N. (1995). *Current status of research regarding to the duration of phenophases and production of autumn rape depending on the variety grown and sowing era.* (Ph.D. dissertation). University of Agricultural Sciences, Iaşi, Romania.
Bland, J. M., & Altman, D. G. (1996). Statistics notes: Measurement error. *BMJ*, *312*(7047), 1654.
Bock, H. H. (2002). *Data mining tasks and methods: Classification: The goal of classification, handbook of data mining and knowledge discovery* (pp. 254–258). New York, NY: Oxford University Press.
Brown, J. D. (2009). Choosing the right number of components or factors in PCA and EFA. *Shiken: JALT Testing & Evaluation SIG Newsletter*, *13*(2), 19–23.
Browne, M. W. (1968). A comparison of factor analytic techniques. *Psychometrika*, *33*(3), 267–334.
Cattell, R. B. (1966). The scree test for the number of factors. *Multivariate Behavioral Research*, *1*(2), 245–276.
Cattell, R. B., & Jaspers, J. A. (1967). A general plasmode for factor analytic exercises and research. *Multivariate Behavioral Research Monographs*, *3*, 1–212.
Chakravarti, I. M., Laha, R. G., & Roy, J. (1967). *Handbook of methods of applied statistics* (Vol. I, pp. 392–394). New York, US: John Wiley & Sons.
Chinchuluun, R., Lee, W. S., Bhorania, J., & Pardalos, P. M. (2009). Clustering and classification algorithms in food and agricultural applications: A survey. In *Advances in Modeling Agricultural Systems Springer Optimization and Its Applications* (Vol. 25). Boston, MA: Springer.
Choi, Y., Taylor, J., & Tibshirani, R. (2017). Selecting the number of principal components: Estimation of the true rank of a noisy matrix. *The Annals of Statistics*, *45*(6), 2590–2617.
Cios, K. J., Swiniarski, R. W., Pedrycz, W., & Kurgan, L. (2007). *Data mining: A knowledge discovery approach*. Boston, MA, US: Springer.
Cooley, W. W., & Lohnes, P. R. (1971). *Multivariate data analysis*. New York, NY: Wiley.
Costello, A. B., & Osborne, J. W. (2005). Best practices in exploratory factor analysis: Four recommendations for getting the most from your analysis. *Practical Assessment, Research and Evaluation*, *10*(7), 1–9.
Fabrigar, L. R., Wegener, D. T., MacCallum, R. C., & Strahan, E. J. (1999). Evaluating the use of exploratory factor analysis in psychological research. *Psychological Methods*, *3*(3), 272–299.
Fatema, K., Maznah, W. O. W., & Isa, M. M. (2015). Spatial variation of water quality parameters in a mangrove estuary. *International journal of Environmental Science and Technology*, *12*(6), 2091–2102.
Friendly, M., & Denis, D. (2005). The early origins and development of the scatterplot. *Journal of Theoretical Social Psychology*, *41*(2), 103–130.
Goh, C. H., Nam, H. G., & Park, Y. S. (2003). Stress memory in plants: A negative regulation of stomatal response and transient induction of rd22 gene to light in abscisic acid-entrained Arabidopsis plants. *The Plant Journal*, *36*(2), 240–255.
Gouveia, R. G., Parreira, E., & Martins, I. P. (2005). Autonomic features in cluster headache. Exploratory factor analysis. *The Journal of Headache and Pain*, *6*, 20–23.
Hakstian, A. R., Rogers, W. D., & Cattell, R. B. (1982). The behavior of numbers of factors rules with simulated data. *Multivariate Behavioral Research*, *17*(2), 193–219.
Hall, J., Matos, S., Silvestre, B., & Martin, M. (2011). Managing technological and social uncertainties of innovation: The evolution of Brazilian energy and agriculture. *Technological Forecasting and Social Change*, *78*(7), 1147–1157.
Harman, H. H. (1976). *Modern factor analysis* (3rd ed.). Chicago, IL: University of Chicago Press.
Hayton, J. C., Allen, D. G., & Scarpello, V. (2004). Factor retention decisions in exploratory factor analysis: A tutorial on parallel analysis. *Organizational Research Methods*, *7*(2), 191–205.
He, X., Ai, X., Jing, Y., & Liu, Y. (2016). Partner selection of agricultural products supply chain based on data mining. *Concurrency and Computation: Practice and Experience*, *28*(4), 1246–1256.
Horn, J. L. (1965). A rationale and test for the number of factors in factor analysis. *Psychometrika*, *30*, 179–185.
Iakovidis, D., & Smailis, C. (2012). A semantic model for multimodal data mining in healthcare information systems. *Studies in Health Technology and Informatics*, *180*, 574–578.
Iakovidis, D. K., Maroulis, D. E., & Karkanis, S. A. (2006). An intelligent system for automatic detection of gastrointestinal adenomas in video endoscopy. *Computers in Biology and Medicine*, *36*(10), 1084–1103.
Iantovics, L. B., Dehmer, M., & Emmert-Streib, F. (2018). MetrIntSimil—An accurate and robust metric for comparison of similarity in intelligence of any number of cooperative multiagent systems. *Symmetry*, *10*(48), 1–21.
Iantovics, L. B., Emmert-Streib, F., & Arik, S. (2017). MetrIntMeas a novel metric for measuring the intelligence of a swarm of cooperating agents. *Cognitive Systems Research*, *45*, 17–29.
Iantovics, L. B., Rotar, C., & Niazi, M. A. (2018). MetrIntPair - A novel accurate metric for the comparison of two cooperative multiagent systems intelligence based on paired intelligence measurements. *International Journal of Intelligent Systems*, *33*(3), 463–486.
Ji, Y. S., Chen, J. J., Niu, G., Shang, L., & Dai, X. Y. (2011). Transfer learning via multi-view principal component analysis. *Computer Science and Technology*, *26*(1), 81–98.
Joanes, D. N., & Gill, C. A. (1998). Comparing measures of sample skewness and kurtosis. *Journal of the Royal Statistical Society (Series D): the Statistician.*, *47*(1), 183–189.
Josse, J., & Husson, F. (2012). Selecting the number of components in principal component analysis using cross-validation approximations. *Computational Statistics & Data Analysis*, *56*(6), 1869–1879.
Kaiser, H. F. (1958). The varimax criterion for analytic rotation in factor analysis. *Psychometrika*, *23*(3), 187–200.
Kaiser, H. F. (1960). The application of electronic computers to factor analysis. *Educational and Psychological Measurement*, *20*, 141–151.
Kantardzic, M. (2002). *Data mining: Concepts, models, methods, and algorithms* (1st ed.). New York, USA: IEEE Press & John Wiley.
Kantardzic, M. (2011). *Data mining: Concepts, models, methods, and algorithms* (2nd ed.). New Jersey, US: IEEE Press & John Wiley.
Kazianka, H., & Pilz, J. (2009). A corrected criterion for selecting the optimum number of principal components. *Austrian Journal of Statistics*, *38*(3), 135–150.
Kenney, J. F., & Keeping, E. S. (1962). Ch. 15: Linear regression and correlation. In *Mathematics of statistics* (Vol. 1, 3rd ed., pp. 252–285). Princeton, NJ: Van Nostrand.
Kim, J. O., & Mueller, C. W. (1978a). *Factor analysis: Statistical methods and practical issues Series Quantitative Applications in the Social Sciences* (). Beverly Hills, CA: Sage Publications.
Kim, J. O., & Mueller, C. W. (1978b). *Introduction to factor analysis: What it is and how to do it Series Quantitative Applications in the Social Sciences* (1st ed.). Beverly Hills, CA: Sage Publications.
Kim, S. M., Sim, Y. S., Seo, H. W., & Son, J. D. (2016). Factor analysis of successful job search for engineering college graduates. *ICIC Express Letters*, *10*(7), 1517–1522.
Kumar, A., Singh, D. P., Yadav, Y. P., & Bikram, S. (1998). Association between morphophysiological parameters and seed yield in brassica genotypes. *Cruciferae Newsletter*, *20*, 69–70.
Kutchtova, P., & Vašáak, J. (1998). Dynamics in creation and reduction of generative organ son winter rapeseed. *Rosliny Oleiste*, *19*(2), 437–446.
Lautenschlager, G. J. (1989). A comparison of alternatives to conducting Monte Carlo analyses for determining parallel analysis criteria. *Multivariate Behavioral Research*, *24*, 365–395.
Lawley, D. N., & Maxwell, A. E. (1971). *Factor analysis as a statistical method* (2nd ed.). London, England: Butterworth & Company.
Ledesma, R. D., & Valero-Mora, P. (2007). Determining the number of factors to retain in EFA: An easy-to-use computer program for carrying out parallel analysis. *Practical Assessment, Research and Evaluation*, *12*(2), 1–11.
Lindeman, R. H., Merenda, P. F., & Gold, R. (1980). *Introduction to bivariate and multivariate analysis*. New York, NY: Scott, Foresman, & Co.

Linn, R. L. (1968). A Monte Carlo approach to the number of factors problem. *Psychometrika*, *33*, 37–71.
Majumdar, J., Naraseeyappa, S., & Ankalaki, S. (2017). Analysis of agriculture data using data mining techniques: Application of big data. *Journal of Big Data*, *4*, 20.
Marusteri, M., & Bacarea, V. (2010). Comparing groups for statistical differences: How to choose the right statistical test? *Biochemia Medica*, *20*(1), 15–32.
Minka, T. (2001). Automatic choice of dimensionality for PCA. *Advances in Neural Information Processing Systems*, *13*, 598–604.
Mondal, R. L., Fattah, Q. A., Latif, A., & Chondhury, K. (1996). Influence of growth regulators on the yield, oil content, fatty acid composition and chemical properties of oil of rape seed. *Journal of the Asiatic Society of Bangladesh, Science*, *22*(2), 141–147.
Montvilas, R. (1999). *Investigations of 00-w-oilseed rape development and productivity on sandyloam soils.* (Thesis). Lithuanian Institute of Agriculture, Lithuania.
Morrison, D. (1967). *Multivariate statistical methods*. New York, NY: McGraw-Hill.
Mucherino, A., Papajorgji, P. J., & Pardalos, P. (2009). *Data Mining in Agriculture, series springer optimization and its applications* (Vol. 34). New York, NY: Springer-Verlag.
Mulaik, S. A. (1972). *The foundations of factor analysis*. New York, NY: McGraw Hill.
Osborne, J. W. (2015). What is rotating in exploratory factor analysis? *Practical Assessment, Research & Evaluation*, *20*(2), 1–7.
Ozer, H., Oral, E., & Dogru, U. (1999). Relationship between yield and yield components on currently improved spring rapeseed cultivars. *Journal of Agriculture and Forestry*, *23*(6), 603–607.
Pearson, K. (1895). Notes on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, *58*, 240–242.
Pearson, K. (1895b). Contributions to the mathematical theory of evolution. II. Skew variation in homogeneous material. *Philosophical Transactions of the Royal Society A - Mathematical Physical and Engineering Sciences*, *186*, 343–414.
Popescu, G. V., Noutsos, C., & Popescu, S. C. (2016). Big data in plant science: Resources and data mining tools for plant genomics and proteomics. *Methods in Molecular Biology*, *1415*, 533–547.
Prather, J. C., Lobach, D. F., Goodwin, L. K., Hales, J. W., Hage, M. L., & Hammond, W. E. (1997). Medical data mining: Knowledge discovery in a clinical data warehouse. *Proceedings of the AMIA Annual Fall Symposium*, 101–105.
Preacher, K. J., Zhang, G., Kim, C., & Mels, G. (2013). Choosing the optimal number of factors in exploratory factor analysis: A model selection perspective. *Multivariate Behavioral Research*, *48*, 28–56.
Raîche, G., Walls, T. A., Magis, D., Riopel, M., & Blais, J. G. (2013). Non-graphical solutions for Cattell's scree test. *Methodology*, *9*(1), 23–29.
Razali, N., & Wah, Y. B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. *Journal of Statistical Modeling and Analytics*, *2*(1), 21–33.
Ro, Y., & Ha, Y. (2019). A factor analysis of consumer expectations for autonomous cars. *The Journal of Computer Information Systems*, *59*(1), 52–60.
Rosner, B. (1983). Percentage points for a generalized ESD many-outlier procedure. *Technometrics*, *25*(2), 165–172.
Rousseeuw, P. J., & Hubert, M. (2011). Robust statistics for outlier detection. *WIREs Data Mining and Knowledge Discovery*, *1*(1), 73–79.
Savchenko, A. V., & Belova, N. S. (2015). Statistical testing of segment homogeneity in classification of piecewise–Regular objects. *International Journal of Applied Mathematics and Computer Science*, *25*(4), 915–925.
Shreck, A., Getz, C., & Feenstra, G. (2006). Social sustainability, farm labor, and organic agriculture: Findings from an exploratory analysis. *Agriculture and Human Values*, *23*(4), 439–449.
Soltis, P. S., Nelson, G., & James, S. A. (2018). Green digitization: Online botanical collections data answering real-world questions. *Applications in Plant Sciences*, *6*(2), e1028.
Song, J., & Belin, T. R. (2008). Choosing an appropriate number of factors in factor analysis with incomplete data. *Computational Statistics & Data Analysis*, *52*(7), 3560–3569.
Stephens, M. A. (1974). EDF statistics for goodness of fit and some comparisons. *Journal of the American Statistical Association*, *69*, 730–737.
Stevens, J. (1986). *Applied multivariate statistics for the social sciences*. Hillsdale, NJ: Erlbaum.
Stigler, S. M. (1989). Francis Galton's account of the invention of correlation. *Statistical Science*, *4*(2), 73–79.
Su, X., & Tsai, C. L. (2011). Outlier detection. *WIREs Data Mining And Knowledge Discovery*, *1*(3), 261–268.
Taylor, J. E., Loftus, J., & Tibshirani, R. J. (2016). Tests in adaptive regression via the Kac-Rice formula. *The Annals of Statistics*, *44*(2), 743–770.
Thurstone, L. L. (1931). Multiple factor analysis. *Psychological Review*, *38*(5), 406–427.
Tietjen, G., & Moore, R. (1972). Some Grubbs-type statistics for the detection of several outliers. *Technometrics*, *14*(3), 583–597.
Treiblmaier, H., & Pinterits, A. (2010). Developing metrics for web sites. *The Journal of Computer Information Systems*, *50*(3), 1–10.
Trewavas, A. (2002). Mindless mastery. *Nature*, *415*(6874), 841.
Trewavas, A. (2005). Green plants as intelligent organisms. *Trends in Plant Science*, *10*(9), 413–419.
Tucker, L. R., Koopman, R. F., & Linn, R. L. (1969). Evaluation of factor analytic research procedures by means of simulated correlation matrices. *Psychometrika*, *34*, 421–459.
Tufféry, S. (2011). *Data mining and statistics for decision making*. Wiley, Chichester, UK: John Wiley & Sons, Ltd.
Velicer, W. F., Eaton, C. A., & Fava, J. L. (2000). Construct explication through factor or component analysis: A review and evaluation of alternative procedures for determining the number of factors or components. In R. D. Goffin & E. Helmes (Eds.), *Problems and solutions in human assessment: Honoring Douglas N. Jackson at seventy* (pp. 41–71). Boston, MA: Kluwer Academic Publishers.
Volkov, A. G., Carrell, H., Baldwin, A., & Markin, V. S. (2009). Electrical memory in Venus flytrap. *Bioelectrochemistry*, *75*(2), 142–147.
Watkins, M. W. (2000). *Monte Carlo PCA for parallel analysis [computer software]*. State College, PA: Ed & Psych Associates.
Watkins, M. W. (2006). Determining parallel analysis criteria. *Journal of Modern Applied Statistical Methods*, *5*(2), 344–346.
Wedel, M., & Shi, W. (2010). Exploratory factor analysis. *Wiley International Encyclopedia of Marketing.*, 2.
Wherry, R. J. (1984). *Contributions to correlational analysis*. New York, NY: Academic Press.
Wichai, D., & Kessomboon, P. (2015). Factors associated with Farmer's behaviors of using pesticide safely: Exploratory factor analysis. *International Journal of Applied Psychology*, *5*(1), 1–7.
Wilk, M. B., & Gnanadesikan, R. (1968). Probability plotting methods for the analysis of data, *Biometrika*. *Biometrika Trust*, *55*(1), 1–17.
Yoneya, K., & Takabayashi, J. (2014). Plant–plant communication mediated by airborne signals: Ecological and plant physiological perspectives. *Plant Biotechnology*, *31*(5), 409–416.
Zamfirescu, N. (1977). *The biological Bases of crop production*. Bucharest, Romania: Ceres Publisher.
Zar, J. H. (1984). *Biostatistical analysis* (2nd ed.). Englewood Cliffs, New Jersey, USA: Prentice-Hall.
Zwick, W. R., & Velicer, W. F. (1986). Comparison of five rules for determining the number of components to retain. *Psychological Bulletin*, *99*(3), 432–442.

**How to cite this article:** Iantovics LB, Rotar C, Morar F. Survey on establishing the optimal number of factors in exploratory factor analysis applied to data mining. *WIREs Data Mining Knowl Discov*. 2018;e1294. https://doi.org/10.1002/widm.1294

Applied Intelligence
https://doi.org/10.1007/s10489-019-01440-5

# MeasApplInt - a novel intelligence metric for choosing the computing systems able to solve real-life problems with a high intelligence

László Barna Iantovics[1] · László Kovács[2] · Corina Rotar[3]



**Abstract**
Intelligent agent-based systems are applied for many real-life difficult problem-solving tasks in domains like transport and healthcare. In the case of many classes of real-life difficult problems, it is important to make an efficient selection of the computing systems that are able to solve the problems very intelligently. The selection of the appropriate computing systems should be based on an intelligence metric that is able to measure the systems intelligence for real-life problem solving. In this paper, we propose a novel universal metric called *MeasApplInt* able to measure and compare the real-life problem solving machine intelligence of cooperative multiagent systems (CMASs). Based on their measured intelligence levels, two studied CMASs can be classified to the same or to different classes of intelligence. *MeasApplInt* is compared with a recent state-of-the-art metric called *MetrIntPair*. The comparison was based on the same principle of difficult problem-solving intelligence and the same pairwise/matched problem-solving intelligence evaluations. Our analysis shows that the main advantage of *MeasApplInt* versus the compared metric, is its robustness. For evaluation purposes, we performed an illustrative case study considering two CMASs composed of simple reactive agents providing problem-solving intelligence at the systems' level. The two CMASs have been designed for solving an NP-hard problem with many applications in the standard, modified and generalized formulation. The conclusion of the case study, using the *MeasApplInt* metric, is that the studied CMASs have the same real-life problems solving intelligence level. An additional experimental evaluation of the proposed metric is attached as an Appendix.

**Keywords** Applied machine intelligence · Computational-hard real-life problem · Cooperative multiagent system · Intelligent system · Machine intelligence · Machine intelligence measure · Real-life problem-solving intelligence

## 1 Introduction

Intelligent agent-based systems (IBASs) consisting of intelligent agents (IAs) and intelligent cooperative multiagent systems (ICMASs) were applied to a large variety of real-life problem solving. In [93], a cooperative multiagent system (CMAS) is presented that is composed of very simple cooperating mobile agents able to perform intelligently network administration tasks, similar to a human network administrator. The paper [46] presents a research on CMAS composed of simple efficiently and flexibly cooperating agents, having at the system's level an increased problem-solving intelligence. There are many other studies that present systems composed of simple cooperating agents, and show that these systems have an emergently increased real-life problem-solving intelligence. This is somehow similar to many simple cooperating life-forms on earth where the intelligence emerges at the cooperating systems' level. Colonies/swarms of biological [48, 75] ants, termites, bugs, and bees are illustrative examples in this sense.

When intelligence estimation is based just on some intuitive principles, as there are presented in many studies, the conclusions are not convincing. Very few metrics were designed worldwide for measuring the machine's intelligence. Each of these metrics is based on some kind of principles of measuring the machine intelligence. The

**Electronic supplementary material** The online version of this article (https://doi.org/10.1007/s10489-019-01440-5) contains supplementary material, which is available to authorized users.

✉ László Barna Iantovics
ibarna@science.upm.ro

Extended author information available on the last page of the article.

application of different intelligence approaches do not allow the direct comparison of the metrics. There is no standardization or even a relatively general view on what an intelligence metric should measure. There is an actual need to design universal metrics for effective measuring of CMASs intelligence. We assume that the artificial systems intelligence should be based on the principle of real-life problem-solving ability. Such an evaluation measure must allow the comparison of CMASs based on real-life problem-solving intelligence. In our study, we concluded that intelligent systems have significant variability in the problem-solving intelligence. This is similar to the variability in the intelligence of the humans.

In recent research performed in 2018 [45] the *MetrInt-Pair* metric, was presented, which was applied to measure the real-life problem-solving machine intelligence of two selected CMASs. The paper also makes an accurate comparison of the intelligence level of the studied CMASs, and at the same time verifies if they can be included in the same class of intelligence (solve problems with the same intelligence). The intelligence comparison of two CMASs is based on some kind of pairwise/matched problem-solving intelligence evaluations.

In this paper, we propose a novel metric called *Robust Metric for Comparison of two Cooperative Multiagent Systems Real-Life Problem-Solving Intelligence* (*MeasApplInt*) that is able to make a simultaneous measuring and comparison of two studied CMASs intelligence. Based on the problem solving intelligence comparison result the two studied CMASs can be classified in the same class of intelligence or in different intelligence classes. The method takes into account also the variability in the intelligence of the compared CMASs. *MeasApplInt* represents an adaptation of the *MetrIntPair* metric providing a more robust behavior. *MeasApplInt* is based on the same principle of pairwise (matched) intelligence evaluations and comparison as *MetrIntPair* metric. The novelty of *MeasApplInt* is based on different performed processings and analyses. This topic is treated in the discussion section. In the case of each compared CMAS, a so-called *Central Intelligence Tendency Indicator* (*CentrIntTen*) is calculated, which quantifies its central intelligence tendency. *MeasApplInt* is also able to verify (by incorporating some calculus in this sense) the indirect correlation in the problem-solving intelligence behavior of the studied CMASs. We will present this subject in the discussions section. *MeasApplInt* is a universal metric of measuring the machine intelligence. Based on this fact its application does not depend on special aspects, like the studied CMASs architecture. It can be applied even for the comparison of two individual agents' problem solving intelligence.

For proving the effectiveness of the proposed *MeasApplInt* metric and for validation purposes two CMASs were studied in Section 4. These two CMASs operated as a *Rank-Based Ant System* [12, 67] and *Min-Max Ant System* [67, 83, 84], specialized in solving an NP-hard problem - the *Travelling Salesman Problem* (*TSP*) [21–23]. The most frequent applications of the *TSP* include the transport, logistics, planning, and the manufacture of microchips. Additionally, it was performed an experimental comparison of the *MetrInt-Pair* and *MeasApplInt* metrics. Another experimental study where intelligence of the studied systems is different is attached as Supplementary Material to this paper.

The upcoming part of the paper is organized as follows: Section 2 analyzes the real-life problem-solving machine intelligence, In Section 2.1 some intelligent systems able to solve different real-life problems are presented, Section 2.2 presents a survey on some significant metrics that are proposed for measuring artificial systems intelligence; In Section 3, our proposed *MeasApplInt* metric for intelligence measuring, comparison, and classification is presented; For validation and comparison purposes, experimental studies are presented in Section 4; Section 5 discusses the proposed metric; In Section 6 the conclusions of the research are presented; The Supplementary Material presents a performed additional experimental study.

## 2 Real-life problem-solving machine intelligence

### 2.1 Intelligent systems applied for real-life problems solving

Intelligent agent-based systems were applied to a large variety of real-life problems solving. Yager [92] proposed an intelligent agent specialized to help in the determination of the appropriateness of displaying a given advertisement to a visitor of a webpage on the WWW. A four layers fuzzy multiagent system able to perform stock price prediction is proposed by Zarandi et al. [90]. Arif et al. [2] proposed assistant agents specialized in e-learning able to help the users to collect different important materials, to examine, and distribute customized knowledge. An agent-based prototype simulation system of public health emergency management is proposed by Song et al. [81]. Iqbal et al. [47] analyzed the necessity and advantages of the use of intelligent agents for different problems that appear in healthcare. The applicability of intelligent agents in E-commerce is analyzed by Kafali and Yolum [50]. Chliaoutakis and Chalkiadakis [15] studied different research questions about early human societies using agent-based simulation. An agent-based model to study compliance with safety regulations of an airline ground service organization is proposed by Sharpanskykh and Haest [78]. A recent overview of autonomous intelligent

vehicle systems that frequently are agent-based by Tokody et al. [85] is realized. Chouhan and Niyogi [18] proposed a domain independent solution based on multiagent systems called MAPJA, for solving planning problems. The problem of scheduling patients in a hospital using multiagent systems by Hsieh [43] is studied. Coelho et al. [16] present an agent-based simulator named MASE-BDI for studying environmental land change.

The papers cited in this section, and many others published in the recent scientific literature, proves the ability of IBASs to solve difficult problems. This motivated the necessity of measuring the machine intelligence. The large variety of IBASs by different architecture intuitively illustrates the difficulty of measuring the machine intelligence, comparing the systems based on their intelligence and classifying them in intelligence classes.

## 2.2 Measuring real-life problem-solving intelligence of agent-based systems

In the previous section, there were presented some representative intelligent agent-based systems specialized in different real-life problems solving. In the presented studies and researches the IBASs are considered intelligent based on different principles. Is not enough to consider a system specialized in real-life problems solving intelligent based just on some principles, without giving a quantitative measure to its intelligence. In this section, we will analyze some metrics proposed for measuring the machine intelligence.

Alan Turing in 1950 [86] gave an early, very interesting definition to the machine intelligence. Turing considered a computing system as intelligent if a human assessor could not decide the nature of the system (being human or otherwise) based on questions asked from a hidden room. Some interesting frequently cited human-computing systems competitions claim the design of such adapted metrics. There are well-known, the competitions between the chess-playing machine named Deep Blue and the chess grandmaster Garry Kasparov [61] and the competitions between the IBM Watson computing system and human experts [36] in the classic game show with a twist named Jeopardy.

In [73] the necessity of creating standard metrics for intelligent systems is outlined. The presented study was realized with the purpose to analyze how precisely intelligent systems are defined and how to measure and compare the capabilities that intelligent systems should provide.

A novel introductory methodology to test the agent's intelligence is presented in [37]. The proposal is based on the calculus of a so-called general intelligence factor and the well-known theory of multiple human intelligences. According to the proposal, an agent uses some principles of the theory of evolution, detaining some stored knowledge. A set of tests assess the multiple intelligences of the agents, for analyzing their behavior in a larger variety of situations. The proposal is intended for both quantitative and qualitative evaluations of intelligence.

In [89] two methods to measure the intelligence of a machine are proposed. In the first step of the research methodology, there were considered to be systems that are called intelligent in the literature. From these systems, there were extracted four common constructs, each of which characterized by several variables. Using this there were suggested two models, represented as entities in three-dimensional construct space. In order to calculate the machine intelligence quotient (MIQ), the Sugeno fuzzy integral and the Choquet fuzzy integral are used. For validation purpose, two applications have been studied for the models using the two fuzzy integrals and are presented some comparative comments.

The idea of elaboration of intelligence test able to be applied to humans and computing systems is analyzed in different studies. In [74] a computer program tested on some standard human IQ tests is presented. According to the authors [74], the computer program surpassed the average human intelligence score by 100 on some tests.

In some studies, the agents' intelligence is considered based on the difficulty/complexity of problems that they are able to solve. In [1] the agent-based systems intelligence considering the ability to compare alternatives based on their complexity is defined. In the performed research, it is considered an agent-based distributed sensor network system. For measuring the intelligence of the system, a specific novel approach is proposed. The proposal was tested by comparing the intelligence quotient in different agent-based scenarios.

In [42] a novel metric for intelligence measuring is proposed. It is based on the theory of the hierarchy of sets of increasingly difficult environments. Hibbard considers an agent's intelligence proportional to the ordinal of the most difficult set of environments that may pass. The measurement tests were implemented with infinite state machine models of computing.

The variety of intelligent swarm systems applied to different problems solving is very wide. The collective intelligence of a particle swarm system according to a novel Maturity Model is assessed in [88]. The proposed model is based on the Maturity Model of Command and Control operational space and on the model of Collaborating Software. The main aim of the study was to obtain a more thorough explanation of how the intelligent behavior of the particle swarm emerges.

In [41] the possibility to design metrics for measuring the capabilities of the cognitive systems is studied. The

conclusion of the study performed by the authors is that a promising approach for measuring the intelligence requires the design of metrics inspired by the human psychometrics that are designed to measure the human cognitive capabilities.

Some studies introduce the notion of universal intelligence test to humans, animals and machines in an integrated way. A comprehensive analysis of the possibility to design universal intelligence tests is performed in [33]. In [40] the idea of a general test called universal anytime intelligence test is studied. In the mentioned study it is considered that such a test must be universal as it is able to measure the intelligence quotient, even if it is very low or very high. The authors also consider that the test should be able to measure the intelligence quotient of any biological or artificial system. The proposal is based on the C-tests and compression-enhanced Turing tests designed in the late 1990s. In their study, the researchers performed an analysis of different tests by highlighting their limitations. They introduce some novel ideas, considering them as necessary for the development of a "universal intelligence test".

In [7] an intelligent system is considered, which can successfully solve difficult problems for humans, has some kind of human-level intelligence. In the paper, difficult problems are studied for the humans that could be used as benchmark problems for intelligent systems.

An interesting study related to the intelligence of cooperative agent coalitions is presented in [17]. The authors propose a novel metric that they consider universal, appropriate to empirically measure the intelligence of different classes of agents. There are presented different illustrative situations where a cooperative multiagent system can be more intelligent than others. There are discussed some common factors that influence the level of collective intelligence.

In [44] a novel metric called *MetrIntMeas* for measuring the intelligence of a cooperative swarm system is proposed. *MetrIntMeas* is able also to make a classification of the studied cooperative swarm system, by verifying if it belongs to the class of cooperative swarm systems with a specific reference machine intelligence. There is also given a definition to the evolution in the intelligence of the cooperative swarm systems.

The identification of the systems that have outstanding (significantly high) or silly (significantly low) intelligence from a set of intelligent systems is analyzed in [3]. For the identification of such systems, there is proposed a specific method called *OutIntSys*. *OutIntSys* can be applied in choosing the systems with outstanding or silly intelligence from a set of intelligent systems able to solve difficult problems.

We made a comprehensive review of the scientific literature focused on the subject of measuring the machine intelligence. This section surveyed some of the research papers that were considered relevant. In some studies, different analysis and evaluations of the systems' intelligence are treated. Metrics that are described in the scientific literature, are designed for making different kinds of measurements of the systems' intelligence.

As a general conclusion, based also on the analogy with the nature/biology, we formulate that there are many studies [19] that treat different aspects of human intelligence but, none of them given a precise definition what the human intelligence is. An interesting fact is that, even in this situation, it is possible to measure the human intelligence (to obtain a quantitative evaluation of the intelligence). There are many designed tests for measuring the human Intelligence Quotient (IQ), see for instance [60]. We consider that a human IQ measure cannot be considered as an absolute indicator but it has practical applicability. For instance, it has some kind of contribution to the job performance along with the detained specialty knowledge specific to the job and sometimes to the detained commonsense knowledge (things that are expected for many peoples to know). Intelligence could be helpful even in more efficient solving of usual every day tasks. Similarly to the humans' intelligence, we consider that the intelligence of the artificial systems cannot be uniquely defined, but even in this context, metrics/tests can be designed for measuring machine intelligence and these metrics if are based on principles of problem-solving ability have applicative utility. Obtained problem solving intelligence measurements could be taken into consideration in decisions related to the choosing of the most intelligent system(s) able to solve specific type(s) of problem(s).

An intelligent agent-based system for solving a specific type of problem could use different architectures. This motivates the necessity to use universal intelligence metrics for measuring machine intelligence. Many designed metrics are based on different principles and they are dependent on some aspects like the IBASs architecture. Based on the limited universality their effective practical utilization is limited. In our approach, we consider the notion universality consisting of the ability to measure machine intelligence no matter what architecture the IBASs has. We do not intend to design a metric that is able to measure artificial and biological intelligence in the same time. This consideration is based on the fact that artificial and natural/biological intelligence are different. The biological intelligence is the result of a very long evolution of life on earth [70]. According to some studies [70], the earth formed about 4.5 billion years ago and some evidence suggests that life emerged prior to 3.7 billion years.

During our studies, we identified that the only important property of the intelligence metrics [3, 44, 45, 62] is to provide a numerical intelligence measure and also to

compare and classify systems based on their intelligence. We considered general processing steps for a metric, as follows: first step, making of some kind of problem-solving intelligence evaluations; second step, performing of some calculus and analyses based on the problem-solving intelligence evaluation measure obtained at the previous step. This procedure allows establishing the intelligence level of the considered system. This general approach in two steps could offer as principal advantages to a metric. In this sense, in [45] an accurate metric called *MetrIntPair* that is able to analyze simultaneously two systems' problems solving intelligence is proposed. An improvement of this metric can be made by increasing its robustness. We consider that this robustness property is necessary in case of many IBASs that are specialized in solving difficult real-life problems.

## 3 MeasApplInt metric for measuring the real-life problem solving intelligence

In this section, a novel universal metric for a robust comparison of the real-life problem-solving intelligence of two studied CMASs is proposed. The metric is described as an algorithm called in the following *Robust Metric for Comparison of two Cooperative Multiagent Systems Real-Life Problem-Solving Intelligence* (*MeasApplInt*).

### 3.1 Type of intelligence, intelligence indicator, central intelligence tendency, intelligence component, and human evaluator

In the following, we determine the intelligence aspects of CMASs.

A *type of intelligence* of a studied CMAS indicates a specific way on that it is able to solve one or more type(s) of problem(s) based on a specific problem-solving point of view appreciated by a *Human Evaluator (HE)*.

An *intelligence indicator* measure obtained at an experimental problem-solving intelligence evaluation gives a quantitative value to the problem-solving intelligence by the considered type.

The *Central Intelligence Tendency* of a studied CMAS represents a quantitative measure given to the studied system's intelligence using a calculus that is based on more experimental difficult problem-solving intelligence evaluation results. It must be mentioned that the *Central Intelligence Tendency* is not an absolute indicator. Its effective obtained numerical value could be slightly different at different sets of experimental problem-solving intelligence evaluations performed. This phenomenon is explained by the variability of machine intelligence of the studied system.

A studied CMAS could have one or more types of problem-solving intelligence. A type of machine intelligence that does not require a formal definition to be given and is not dependent on the architecture of the intelligent system. The human evaluator must identify based on the type(s) of problem(s) that the studied intelligent system should solve the type of problem-solving intelligence by interest.

There are different studies, researches and even patents related to specialists that act as human evaluator. In [71] a methodology for analysis and classification of different kind of human error is proposed. The methodology treats the possible causes and factors that contribute to the occurrence of different errors. The US Patent [6] is focused on a general approach to some methods for presentation and evaluation of constructed responses assessed by human evaluators. In [58], investigates the contribution of human evaluator in image enhancement tasks performed in order to make visual improvements of images. The study presented in the paper [49] investigates the perceived value of two sentiment analysis tools developed to understand Finnish language, in contrast to human evaluators. A research in [13] added the human contributor to the mechanized system of evaluation in an e-Learning environment. In the paper [80], the central role of human evaluators is analyzed in the applied automatic semantic technologies.

The studies briefly discussed in the previous paragraph present the important role that could play a human evaluator in different studies and problem-solving based on the detained knowledge and experience. In our research, the *HE* is a specialist who must detain knowledge related to the studied intelligent systems and the type(s) of problems that they must solve. A decision process of the *HE* practically includes the identification of type of the intelligence by interest and how to measure it. The corresponding principle to a type of intelligence is practically a verbal explanation associated to it. In case of systems with one or a very low number of types of intelligence, the deciding process of *HE* could be simple. In the case of extremely complex systems with higher number of types of intelligences, the decision processes could be highly complex, and *HE* could be represented by a team of specialists.

The decision related to the identification of types of intelligence that an intelligent system may possess can be improved if it is made automatically. The automatic identification of the types of intelligences that a system could detain represents our next research direction. It is proposed the study of the design of a mathematical model for the automatic identification of the types of intelligences that an intelligent system possesses.

**An illustrative scenario to the previously introduced notions**

*TSP* applications solve transportation problems that can be

defined as the movement of humans and goods from one location to another. In order to illustrate the previously introduced notions, in the following, a scenario is presented. It is considered an intelligent next-generation agent-based flying drone. The flying drone specialization consists in the delivery at a journey of a set of products to one or more destinations (at home to the clients) and to return to the starting point. Each destination must be visited a single time, and it includes the delivery of one or more products. *HE* can simply consider the machine intelligence based on the intelligence in the performing of an efficient low cost of delivery. According to this setting, the lower the cost the higher is machine intelligence. The cost could include among others, the fuel consumption necessary to deliver the set of the products by interest to one or more clients. The fuel consumption is dependent on aspects like the traveling distance, the weight of the products that must be transported on different distances and the performed departures and landings. The cost may depend also on the aspects like the meteorological conditions (there is a storm in a specific region and the drone must avoid that region for example).

If necessary, in case of a very complex CMAS based on its specificity, the intelligence indicator can be calculated as the weighted sum of some *intelligence components measure*, which gives a quantitative indication to different aspects of the system's intelligence.

Let us denote with $Ind_r$ a type of intelligence indicator chosen for a CMAS, $Ind_r = wg_1 \times ms_{r,1} + wg_2 \times ms_{r,2} + \ldots + wg_p \times ms_{r,p}$; $wg_1 + wg_2 + \ldots ... + wg_p = 1$.$Ind_r$ is calculated as the weighted sum of $p$ types of intelligence components measure, components denoted with $1, 2, \ldots, p$. $ms_{r,1}, ms_{r,2}, \ldots, ms_{r,p}$ represent the considered intelligence components' measures obtained at a problem-solving intelligence's evaluation, and $wg_1$, $wg_2, \ldots, wg_p$ represent the components weights.

In the case of the previously presented flying drone scenario, if the intelligence is seen in a complex way, then, an intelligence component established by the human specialist could for instance:

1. Offer a quantitative measure of the problem-solving intelligence based on the efficiency of delivery. This could include aspects, like the total time of delivery, the fuel consumption necessary for the delivery, satisfaction of the clients related to the quality of delivery;
2. Offer a quantitative measure to the cooperation efficiency with other flying drones. For example, cooperation performed in order to avoid the collisions during the flight with other similar flying drones;
3. Offer a quantitative measure to the capacity to avoid different types of flying entities that are unable to make an intelligible communication in order of their avoidance, like the birds, for example.

In the case of the intelligence indicator's calculus as multiple intelligence components measure, sometimes a transformation of the components must be performed before making the performed calculus of the experimental problem-solving intelligence measure.

**With explanatory purposes, in the following we present some scenarios** In order to illustrate the idea of weight of an intelligence component, we consider the scenario when the flying drone intelligence at each flight is calculated using two intelligence components measure established by the *HE*, based on the intelligence of delivery.

*Scenario 1* At a problem-solving (delivery of one or more product to one or more clients at home) a mark between 1 and 10 is given by a human specialist in intelligent systems. This mark (the first intelligent component measure) is considered to have the weight by 0.6. The second mark (the second intelligence component measure) is established as the average of the marks with values between 1 and 10 given by all the clients who receive products at that specific delivery with the considered weight by 0.4. For example, the problem of delivery of some products to three clients is considered. Each client gives a mark and the average of the values is calculated. The weight of this average is marked with 0.4. The rationale for choosing these weights by *HE* could be based on the fact that the human specialist has a deep knowledge about intelligent systems, and the clients are usual peoples that give the mark mostly based on the satisfaction of the delivery by the drone, but most of them do not have any knowledge about intelligent systems. The rationale of the fact that the weight of the evaluation of clients is close to the weight of the evaluation of the specialist could be based on the aspect that clients' satisfaction of delivery is very important for the company that detains such drones. The final calculated intelligence indicator value based on the values of intelligence components will be a number in the interval [1, 10], with 1 signifying the no intelligence and 10 signifying highest possible intelligence.

*Scenario 2* We consider the scenario when *HE* evaluates the following two intelligence components, the benefit of the delivery expressed in money with the weight 0.7 and a mark between 1 and 10 given to the intelligence of the drone by a human specialist in intelligent systems, having the weight 0.3. In this case, in order to calculate the value of intelligence indicator, a mathematical transformation of the value of benefit into a numerical value in interval [1, 10] must be applied. The resulting intelligence of the flying drone at a problem solving, the obtained intelligence indicator value will be a number in the interval [1, 10]. With 1 signifying no intelligence and 10 signifying the highest possible intelligence.

### 3.2 Description of the proposed *MeasApplInt* metric

In the following, we consider two studied CMASs denoted as $COP^{i1}$ and $COP^{i2}$ that solve the same type/class of real-life problems denoted $ClassP$. The intelligence testing of $COP^{i1}$ and $COP^{i2}$ is considered on the same set of test problems denoted $DifPr = \{DifPr_1, DifPr_2, \ldots, DifPr_k\}$. The obtained results of the intelligence indicator for the problems solving are denoted as $IndSet_1 = \{Ind_1n_1, Ind_1n_2, \ldots, Ind_1n_k\}$ and $IndSet_2 = \{Ind_2n_1, Ind_2n_2, \ldots, Ind_2n_k\}$. $|IndSet_1|$ and $|IndSet_2|$ represent the cardinality (sample size) of $IndSet_1$ and $IndSet_2$, where $|IndSet_1| = |IndSet_2| = k$.

Table 1 shows the organization of the results of problem-solving experimental intelligence evaluation for the studied CMASs. Table 1 presents the obtained intelligence indicators' sets for both CMASs. $Ind_1n_1$, $Ind_1n_2 \ldots Ind_1n_k$ - represents the problem-solving intelligence evaluation results of $COP^{i1}$; where $Ind_1n_1$ represents the measured intelligence in the $DifPr_1$ solving, $\ldots Ind_1n_k$ represents the measured intelligence in the $DifPr_k$ solving. $Ind_2n_1$, $Ind_2n_2, \ldots, Ind_2n_k$ - represents the problem-solving intelligence evaluation results of the $COP^{i2}$; where $Ind_2n_1$ represents the measured intelligence in the $DifPr_1$ solving,$\ldots Ind_2n_k$ represents the measured intelligence in the $DifPr_k$ solving.

In the following, the *Central Intelligence Tendency Indicator* (*CentrIntTen*) of a CMAS, is defined as its indicator of the central intelligence tendency in real-life problem-solving. The intelligence of the biological life forms, including the humans, has variability in intelligence that is manifested as oscillations in intelligence. We consider that, similarly to the biological intelligence, the CMASs have variability in intelligence. $CentrInt_1$ and $CentrInt_2$ denote the *CentrIntTens* of $COP^{i1}$ and $COP^{i2}$. We considered the most appropriate *CentrIntTens* of $COP^{i1}$ and $COP^{i2}$ the medians of $IndSet_1$ and $IndSet_2$. $CentrInt_1$ =*Median*($Ind_1n_1$, $Ind_1n_2$, …, $Ind_1n_k$), $CentrInt_2$ =*Median*($Ind_2n_1$, $Ind_2n_2$, …, $Ind_2n_k$). The decision for choosing the median as the indicator of the central intelligence tendency was based on the required robustness, from the measured intelligence indicator data property point of view. There is no restriction to the assumption that $IndSet_1$ and $IndSet_2$ should be normally distributed (sampled from a Gaussian population) and they can be even heterogeneous.

The *MeasApplInt* metric presented in the form of an algorithm called *Robust CMASs Intelligence Measuring, Comparison and Classification* compares the central intelligence tendency of $COP^{i1}$ and $COP^{i2}$ on the testing problem set *DifPr*. Figure 1 contains the graphical representation of the specific processing and analyzing steps performed by the metric. *MeasApplInt* checks if the intelligence of the studied CMASs are equal (there is no statistical difference between them) or different from the statistical point of view. Further, we consider *Null Hypothesis* denoted as $H0^{ie}$, the statement that $COP^{i1}$ and $COP^{i2}$ intelligence are equal from the statistical point of view (more precisely statistical equality of the *CentrIntTens*). We denote with $H1^{ie}$ the *Alternative Hypothesis*, a hypothesis that indicates that the intelligence of $COP^{i1}$ and $COP^{i2}$ is different from the statistical point of view.

*MeasApplInt* uses as input $IndSet_1 = \{Ind_1n_1, Ind_1n_2, \ldots..., Ind_1n_k\}$ and $IndSet_2 = \{Ind_2n_1, Ind_2n_2, \ldots.., Ind_2n_k\}$ that represents the $COP^{i1}$ and $COP^{i2}$ intelligence indicators measure obtained during the studied CMASs evaluation in solving of the test problems set denoted *DifPr*. The cardinality of the problems set is given by |*DifPr*|=*k*. The choosing of the appropriate problems set for the intelligence measurements is the responsibility of *HE* based on the type of intelligence that *HE* would like to detect.

The first step of the *MeasApplInt* algorithm indicates a statistical characterization of the intelligence indicators data by computing the values for the most important statistical indicators [54, 63]: *Mean*; *Standard Error of Mean*

**Table 1** Pairwise intelligence evaluation results of the studied $COP^{i1}$ and $COP^{i2}$

| $PrId^{\wedge}$ | $Problem^{*}$ | $IndSet_1{}^{\#}$ | $IndSet_2{}^{\#\#}$ | $Pairs^{\&}$ |
|---|---|---|---|---|
| *1* | $DifPr_1$ | $Ind_1n_1$ | $Ind_2n_1$ | $Ind_1n_1 - Ind_2n_1$ |
| *2* | $DifPr_2$ | $Ind_1n_2$ | $Ind_2n_2$ | $In_1n_2 - Ind_2n_2$ |
| … | … | … | … | … |
| *K* | $DifPr_k$ | $Ind_1n_k$ | $Ind_2n_k$ | $Ind_1n_k - Ind_2n_k$ |
| | | $CentrInt_1$ | $CentrInt_2$ | |

$^{\wedge}$denotes the problem identifier; $^{*}$denotes the problem used in the problem-solving intelligence evaluation, called independent variable; $^{\#}$denotes the measured problem-solving intelligence of $COP^{i1}$, called dependent variable; $^{\#\#}$ denotes the measured problem-solving intelligence of $COP^{i2}$, called dependent variable; $^{\&}$denotes the formed pairs, with $^{\#}$dependent variable and $^{\#\#}$dependent variable

(*SEM*); *Median*; *Standard Deviation* (*SD*); *Variance*; *Lowest value* (*Min*); *Highest value* (*Max*); *Confidence level of the mean* (*CL*); [*LowerCI*, *UpperCI*] the lower and higher bound of the confidence interval of the mean and *Coefficient of Variation* (*CV*). As *CL* of the confidence interval of the mean in most of the cases, we propose the choosing of 95%. The value of *CV, CV*=100x(*SD/Mean*) is appropriate for analyzing the homogeneity-heterogeneity of a sample intelligence data set. *CV* indicates the variability in intelligence in the problem-solving. We considered the data classification based on the variability as follows [34, 56]: $CV \in [0,10)$ indicates homogeneous data (*hom.*); $CV \in [10,20)$ indicates relatively homogeneous data (*rel-hom.*); $CV \in [20,30)$ indicates relatively heterogeneous data (*rel-het.*); $CV \geq 30$ indicates heterogeneous data (*het.*).

***MeasApplInt:***
***Robust CMASs Intelligence Measuring, Comparison and Classification***
**INPUT:**
$IndSet_1 = \{Ind_1n_1, Ind_1n_2, \ldots, Ind_1n_k\}$; $IndSet_2 = \{Ind_2n_1, Ind_2n_2, \ldots, Ind_2n_k\}$;
**OUTPUT:**
//Calculated *Central Intelligence Tendency Indicators* of $COP^{i1}$ and $COP^{i2}$.
$CentrInt_1$; $CentrInt_2$;
// Decision regarding to the intelligence comparison of $COP^{i1}$ and $COP^{i2}$.
*DecIntell;*
**Begin**
**Step 1.** *Analysis of the intelligence indicators data.*
@Set the *CL* value; @Performs a statistical characterization of $IndSet_1$ and $IndSet_2$ by calculating the:
*Mean*; [*LowerCI*, *UpperCI*]; *SEM*; *SD*; *Variance*; *Min*; *Max*;
@Calculates the *CV* for both $IndSet_1$ and $IndSet_2$. Analyze the homogeneity of $IndSet_1$ and $IndSet_2$ data sets based on the *CV* value;
@Calculates $CentrInt_1$ and $CentrInt_2$ as the medians of $IndSet_1$ and $IndSet_2$;
@Report$CentrInt_1$ and $CentrInt_2$;
**Step 2.** *Verification of the data normality assumption.*
@Verify if $IndSet_1$ and $IndSet_2$ are normally distributed;
**If** (both $IndSet_1$ and $IndSet_2$ are normally distributed) **Then**
//Calculates the secondary *CentrIntTens*.
@Calculates $SecCentrInt_1$ as the mean of $IndSet_1$;
@Calculates $SecCentrInt_2$ as the mean of $IndSet_2$;
@Report $SecCentrInt_1$ and $SecCentrInt_2$;
@Indicates to the *HE* that the proposed *MeasApplInt* metric is applied, but there is a chance to be designed a more appropriate metric;
**EndIf**
**Step 3.** *Verification of the pairing/matching effectiveness.*
@Apply the ***Effective Pairing Verification Algorithm***;
**If** (*PairingEffectiveness*=*"Yes"*) **Then**
*@Report "The pairing was effective"*;
**Else** *@Report "The pairing was not effective"*;
**Endif**
**Step 4.** *Comparison and classification of the studied CMASs intelligence.*
@Set the $\alpha Class$ parameter value;
@Formulate the hypotheses $H0^{ie}$ and $H1^{ie}$;
**If** (*PairingEffectiveness*=*"Yes"*) **Then**
@Apply the nonparametric *Wilcoxon signed rank test* for two paired samples.
Obtains the *PvalClass*.
**Else**
@Apply the nonparametric *Mann-Whitney test*. Obtains the *PvalClass*.
**Endif**
**Step 5.** *Decision regarding the intelligence comparison and classification*
**If** ($PvalClass > \alpha Class$) **Then**
@Accept $H0^{ie}$; //There is no evidence for the $H0^{ie}$ rejection.
//The intelligence of $COP^{i1}$ and $COP^{i2}$ is equal from statistical point of view.
*DecIntell:*="$COP^{i1}$ and $COP^{i2}$ CAN be included in the same class of intelligence.";
**Else**
@Accept $H1^{ie}$;//$H0^{ie}$ is rejected;
//$COP^{i1}$ and $COP^{i2}$ CANNOT be included in the same class of intelligence.
**If** ($CentrInt_1 < CentrInt_2$) **Then**
*DecIntell:*="$COP^{i1}$ *is less intelligent than* $COP^{i2}$";
**Else** *DecIntell:*=" $COP^{i1}$ *is more intelligent than* $COP^{i2}$";
**EndIf**
***EndMeasApplInt***

One of the advantages of the proposed metric consists in the fact that is robust, therefore, it is appropriate in the case of sample data that does not follow the Gaussian distribution even if it is not homogenous or relative-homogeneous. Step 2 of the algorithm, indicates that *MeasApplInt* is applied even in case of Gaussian data, but in this situation, it is reported to *HE* that it is possible (there is a chance) to design a more accurate metric.

We can formulate this affirmation as a general principle in Artificial Intelligence, as follows: if some additional properties are known about a specific problem then, there is some chance to design a more efficient problem-solving method/ algorithm than the methods/ algorithms designed without exploring those additional properties. As a very

simple explanatory example (it is not related to intelligent systems, but it is illustrative to the principle that we formulated), we mention the search for a number in a large set of numbers. If there is no information about the property of the set of numbers, then the sequential search method can be applied. Elsewhere, for example, if the numbers are ordered, based on this important additional property, it can be considered that there is a chance to be designed a more appropriate search method, which takes into consideration the ordering property of the numbers. It is well-known that the binary search method is suitable for this situation and it explores very well the ordering property of the numbers.

If the data sets $IndSet_1$ and $IndSet_2$ are normally distributed, then there are computed the two secondary intelligence indicators $SecCentrInt_1$ and $SecCentrInt_2$ as the means of $IndSet_1$ and $IndSet_2$. In this case, they can be considered as indicators of the secondary central intelligence tendency.

For the verification of the data's normality, we propose the *One-sample Kolmogorov-Smirnov test* (*K-S test*) [14] and the *Shapiro-Wilk test* (*S-W test*) [77]. We propose to use the normality tests at the significance level, $\alpha Norm$=0.05. The Quantile-Quantile plot (Q-Q plot) is a scatterplot appropriate for the visual appreciation of the normality. A Q-Q plot is created by plotting two sets of quantiles, one against another. If both sets of quantiles came from the same distribution, the points form a line that is roughly straight. We suggest the joint use of Q-Q plot with the Shapiro-Wilk test for obtaining of a more accurate conclusion related to the normality (approximated normality, usually there is no perfect normality in case of real-life data).

Step 3 of the *MeasApplInt* algorithm suggests the verification of pairing (matching) effectiveness, using a novel algorithm called *Effective Pairing Verification*, abbreviated as *II-EffPair*. According to the *Effective Pairing Verification* algorithm, if $IndSet_1$ and $IndSet_2$ are normally distributed, the correlation indicator (coefficient) denoted $R$ is calculated as the *Pearson Correlation Coefficient* [38, 65, 82], elsewhere it is computed the *Spearman Correlation Coefficient* [9, 59]. This decision is based on the fact that *Spearman Correlation Coefficient* is more robust than the *Pearson Correlation Coefficient*, and, considering this circumstance, it is more appropriate in the nonparametric case.

If the pairing (matching) is effective it can be decided for the application of the *Wilcoxon* test (*Wilcoxon signed rank test* for two paired samples) that takes into consideration the pairing property. The robust *Wilcoxon* nonparametric test, was proposed by Frank Wilcoxon [68, 69, 79, 87]. If the pairing (matching) assumption is violated, then is indicated the application of the *Mann-Whitney test* [35, 55] instead of *Wilcoxon's test*. The nonparametric *Mann-Whitney test* is similar to the Wilcoxon test from the robustness point of view. The disadvantage of the *Mann-Whitney test* compared to *Wilcoxon's test* consists in the fact that it does not explore the pairing property of intelligence data, as this requires a slightly increased sample size. It is suggested the application of the two-tailed test instead of the one-tailed test in case of any chosen test, *Wilcoxon test* or *Mann-Whitney test*. $\alpha Class$ denotes the significance level of the applied statistical test. $\alpha Class$ represents the probability to make a type I error, to reject $H0^{ie}$ (*Null Hypothesis*) when it is true. As value of $\alpha Class$, $\alpha Class$=0.05 is suggested that we consider it as the most appropriate in most of the situations.

***II-EffPair:***
***Effective Pairing Verification Algorithm***
**INPUT:** //The intelligence sample data sets whose pairing is verified.
$IndSet_1 = \{Ind_1n_1, Ind_1n_2, \ldots, Ind_1n_k\}$; $IndSet_2 = \{Ind_2n_1, Ind_2n_2, \ldots, Ind_2n_k\}$;
**OUTPUT:** *PairingEffectiveness*;//Result of pairing effectiveness.
**Step1.** *Calculus of the correlation coefficient.*
$\alpha Corr := 0.05$; *ConfCorr:=95%;*
**If** ($IndSet_1$ and $IndSet_2$ are normally distributed) **Then**
@Calculates the *Pearson Correlation Coefficient* denoted *R*.
@Performs a statistical test in order to verify if *R* is statistically different from
0. Let *PvalCorr* be the obtained P-value of the statistical test.
@Calculates the confidence interval for *R* with the *ConfCorr* confidence level.
**Else**
@Calculate the *Spearman Correlation Coefficient* denoted *R*.
@Performs a statistical test in order to verify if *R* is statistically different
from 0. Let *PvalCorr* be the obtained P-value of the statistical test.
@Calculates the confidence interval for *R* with the *ConfCorr* confidence level.
**Endif**
**Step2.** *Verification of the pairing/matching effectiveness.*
**If**(*PvalCorr*> $\alpha Corr$) **Then**
@Report *PairingEffectiveness:="No";*
**Else** @*Report PairingEffectiveness:="Yes";*
**Endif**
***EndEffectivePairingVerification***

By applying the *MeasApplInt* metric, if *PvalCorr* > $\alpha$Class, then it can be decided that $H0^{ie}$ is accepted. There is no evidence for the $H0^{ie}$ rejection. This may be formulated as follows, even there is a numerical difference between the computed central intelligence indicators $CentrInt_1$ of $COP^{i1}$ and $CentrInt_2$ of $COP^{i2}$, there is no statistical difference between them, and therefore,

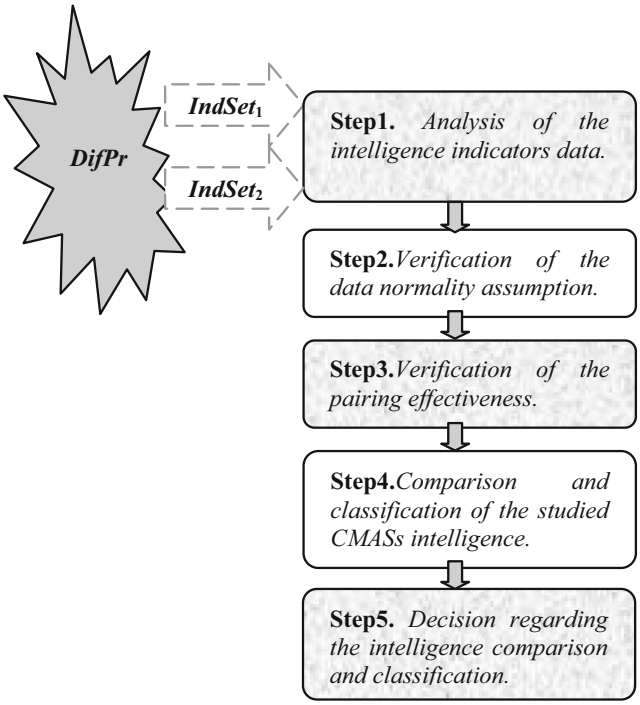



**Fig. 1** The flowchart of data processing in the *MeasApplInt* algorithm

the studied CMASs have the same problem-solving intelligence. The numerical difference between $CentrInt_1$ and $CentrInt_2$ is caused by the variability in the intelligence of the CMASs. From classifications point of view, both CMASs, $COP^{i1}$ and $COP^{i2}$ can be classified in the same class of intelligence.

If $Hl^{ie}$ is accepted, then it can be concluded that the intelligence of $COP^{i1}$ and $COP^{i2}$ is different. The numerical difference between the $CentrInt_1$ and $CentrInt_2$ *CentrIntTens* is statistically significant and it is not the consequence of the variability. From classification point of view, $COP^{i1}$ and $COP^{i2}$ cannot be classified in the same class of intelligence. If $CentrInt_1 < CentrInt_2$, then it can be concluded that $COP^{i1}$ is less intelligent than $COP^{i2}$. If $CentrInt_1 > CentrInt_2$ then it can be concluded that $COP^{i1}$ is more intelligent than $COP^{i2}$.

## 4 Case study for comparison of two CMASs problem-solving intelligence

### 4.1 Real-life applications of the Travelling Salesman Problem

*Travelling Salesman Problem* (*TSP*) can be defined as follows: given a map that includes $K$ cities, a salesman starting from a city, wish to visit each city a single time and then return to home. He/she would like that the total traveling cost (distance, traveling time etc.) to be the smallest possible. *TSP* remains one of the most challenging problems in operational research based on the fact that it is an NP-hard problem (non-deterministic polynomial-time hard) [27, 52]. NP-hardness is the defining property of a class of problems that are, at least as hard as the hardest problems in *NP*. In the case of a class of an NP-hard problem it is unlikely a polynomial-time algorithm for solving will be ever found. One of the earliest formal definitions of *TSP* describes it as an integer linear program [29].

**Definition 1** Given the complete graph $G = (N,\ E)$ where the set of vertices is $N = \{1, 2, \ldots, n\}$, and the positive matrix $C = (c_{ij})_{1 \le i,j \le n}$ where $c_{ij}$ is the cost associated to the edge $(i\,j)$, TSP defines the integer variables $(x_{ij})_{1 \le i,j \le n}$ which correspond to a generic path in $G$

$$x_{ij} = \begin{cases} 1 & if\ the\ edge\ (i\ j)\ is\ used \\ 0 & otherwise \end{cases} \tag{1}$$

and minimizes the objective function (2) subject to constraints (3), (4) and (5).

$$\sum_{i,j=1}^{n} x_{ij} \times c_{ij} \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall\ 1 \le j \le n \tag{3}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall\ 1 \le i \le n \tag{4}$$

$$\sum_{i,j \in S} x_{ij} \le |S| - 1 \quad \forall\ S \subset N, \quad 2 \le |S| \le n - 1 \tag{5}$$

The constraints (3, 4) make sure that only two edges in a selected path are incident to any vertex. The constraint (5) excludes circuits only using vertices from a proper subset of $N$.

*TSP* can be classified in *Symmetric TSP* (*sTSP*) and *Asymmetric TSP* (*aTSP*). *sTSP* [39, 64] is the problem of finding the shortest Hamiltonian cycle/tour in a weighted undirected graph that does not have loops or multiple edges. The distance between two cities is the same in each opposite direction. Many practical combinatorial optimization problems in production management and scheduling can be formulated as equivalent to the *sTSP*.

The *aTSP* [29] characterizes the situation when edges may not exist in both directions or the distances might be different; a directed graph is formed. One-way streets are examples of situations when the symmetry property is not satisfied.

The *Travelling Purchaser Problem* (*TPP*) and the *Vehicle Routing Problem* (*VRP*) are both NP-hard problems that

represent generalizations of *TSP*. The *TPP* can be enounced as follows "Given a list of marketplaces, the cost of travelling between different marketplaces, and a list of available products together with the price of each such product at each marketplace, the task is to find, for a given list of articles, the route with the minimum combined cost of purchases and traveling" [8]. The *VRP* is enounced as follows "What is the optimal set of routes for a fleet of vehicles to traverse in order to deliver to a given set of customers?" [28].

### 4.2 Cooperative multiagent systems that mimic the biological ants

Even very simple creatures like the natural ants, bees, African termite species and some others, which does not have intelligence at the individual level could be considered intelligent at the level of the colony/swarm. There are many studies that prove the emergent real-life problem-solving intelligence of colonies/swarms composed of simple living creatures able to intelligently solve complex tasks [11, 48].

Beni and Wang [5] introduced the notion of Swarm Intelligence (SI) in the context of robotic systems called Cellular Robotic Systems. The use of swarms of simple cooperative mobile robots may have advantages like scalability (the swarm can be extended with new members, inefficient members can be eliminated), robustness (the swarm can solve a problem even if some of its members fail, does not have enough fuel for operation, for example) and problem solving efficiency (distributed cooperative problem solving). Cooperative swarms of ground and aerial vehicles have diverse applications like convoy protection, [76] and moving target localization and tracking [51].

Marco Dorigo [20, 31] firstly proposed the problem-solving based on simple computing agents that mimic the behavior of natural ants in the way they search for the food. There are many applications of the algorithms that mimic the biological ants like, optimization of clustering models [72], emergency management using geographic information systems [27], solving permutation scheduling problems [57] and optimization of the constrained mechanical design [66].

In the following, the general operation of a cooperative multiagent system is briefly presented that operates as an *Ant System* (*AS*). Initially, each agent (artificial ant) is placed on some randomly chosen node (city). An agent $k$ currently at node $i$ chooses to move to node $j$ by applying a probabilistic transition rule (6).

$$p_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)]^{\alpha} \times [\eta_{ij}]^{\beta}}{\sum_{l \in J_k(i)} [\tau_{il}(t)]^{\alpha} \times [\eta_{il}]^{\beta}} & \text{if} \quad j \in J_k(i) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

After each agent completes its tour, the pheromone amount on each path will be adjusted according to (7), (8) and (9).

$$\tau_{ij}(t+1) = (1-\rho) \times \tau_{ij}(t) + \Delta\tau_{ij}(t) \tag{7}$$

$$\Delta\tau_{ij}(t) = \sum_{k=1}^{k=m} \Delta\tau_{ij}^{k}(t) \tag{8}$$

$$\Delta\tau_{ij}^{k}(t) = \begin{cases} \frac{Q}{L_k} & \text{if} \quad \text{(i,j)} \in tour\_performed\_by\_agent\_k \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$\rho$, $\alpha$ and $\beta$ that are used in (6)–(9) are adjustable parameters. $\alpha$ is called the power of the pheromone. $\beta$ is called the power of the distance weight. The role of $\alpha$ and $\beta$ is to control the relative weights of the heuristic visibility and the pheromone trail. $\alpha$ and $\beta$ establish the necessary trade-off between edge length and pheromone intensity. $Q$ is an arbitrary constant. In many studies the value of $Q=1$ is considered as the most appropriate. $d_{i,k}$ represents the distance between the nodes denoted $i$ and $k$. The variables $\eta_{ik}$ stand for the heuristic visibility of the edge $(i,k)$.

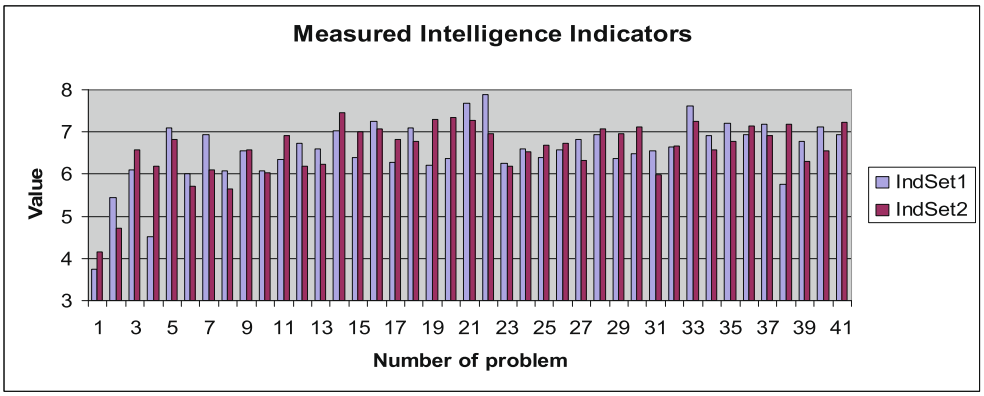



**Fig. 2** Graphical representation of Intelligence Indicators for $COP^{i1}$ and $COP^{i2}$

$\eta_{ik} = 1/d_{i,k}$. $\rho$ represents the evaporation factor. The value of $\rho$ must be in the interval (0, 1). The number of agents that forms the *AS* is denoted by $m$. $L_i$ denotes the length of the tour performed by the agent $i$.

For the validation of the proposed metric, a case study was conducted in order to prove its effectiveness. We considered the comparison of two cooperative multiagent systems intelligence denoted $COP^{i1}$ and $COP^{i2}$, formed by mobile agents, that solves the *sTSP*.

$COP^{i1}$ operated as a *Rank-Based Ant System* (*RBAS*). In the *RBAS* [12, 67], modified version of the *AS* the obtained solutions are ranked according to their length. Further, the amount of deposited pheromones is weighted for each solution. However, solutions with shorter paths deposit more pheromones than the solutions with longer paths.

$COP^{i2}$ operated as a *Min-Max Ant System* (*MMAS*). A *MMAS* [67, 83] is a variant of the *AS*. A *MMAS* give dynamically evolving bounds on the pheromone trail intensities, this is done in such a way that the pheromone intensity on all the paths is always within a specified limit of the path with the greatest pheromone intensity. *MMAS* uses lower and upper pheromone bounds to ensure that all of the pheromone intensities are between the two bounds.

In the experimental setup, there were considered maps composed of *nr*=30 randomly placed cities. The parameters of both $COP^{i1}$ and $COP^{i2}$ were considered: *repetitions*=1200; $\alpha$=1.738; $\beta$=2.085 and $\rho$=0.163. $|DifPr|$=41. $DifPr = \{DifPr_1, DifPr_2,...,DifPr_{41}\}$. The intelligence indicator for both *CMASs* was considered as the obtained best-to-date travel value from the beginning of the problem-solving process. Figure 2 contains the graphical representation of the variability of intelligence indicators data $IndSet_1$ and $IndSet_2$. Table 2 presents the obtained $IndSet_1$ and $IndSet_2$ intelligence evaluation's results for $COP^{i1}$ and $COP^{i2}$.

Table 3 presents the results of the performed statistical characterization as indicated in the first step of the *MeasApplInt* algorithm. *CL* was set to 95%. The *CV* values of $IndSet_1$ and $IndSet_2$ indicate a moderate relative homogeneity (with values between 10 and 12).

Table 4 presents the results of the performed normality test of $IndSet_1$ and $IndSet_2$. The data normality was verified with the One-sample Kolmogorov-Smirnov test and Shapiro-Wilk test. We choose to apply the two normality tests at the significance level, $\alpha Norm$=0.05. Figures 3 and 4 presents the Normal Q-Q Plot of $IndSet_1$ and $IndSet_2$, for the visual analysis of the normality, which must be used jointly for the accurate normality estimation with the Shapiro-Wilk test. *PvalNorm* denotes the P-value of the normality test. $PvalNorm > \alpha Norm$ should be interpreted as the normality is verified, elsewhere the normality assumption failed. The obtained test results presented in Table 4, and the visual representation of Figs. 3 and 4, shows that none of the intelligence indicator data sets $IndSet_1$ and $IndSet_2$ passed the normality assumption.

Step 3 of the *MeasApplInt* algorithm indicates the verification of the effective pairing (matching) property of

**Table 2** Intelligence evaluation results of the $COP^{i1}$ and $COP^{i2}$

| $IndSet_1$ | $IndSet_2$ |
|---|---|
| $Ind_1n_1$[3.752]*[1] | $Ind_2n_1$[4.159]#[1] |
| $Ind_1n_2$[5.442]#[3] | $Ind_2n_2$[4.717]*[3] |
| $Ind_1n_3$[6.106] | $Ind_2n_3$[6.579] |
| $Ind_1n_4$[4.513]*[2] | $Ind_2n_4$[6.189]#[2] |
| $Ind_1n_5$[7.084] | $Ind_2n_5$[6.831] |
| $Ind_1n_6$[6.012] | $Ind_2n_6$[5.707] |
| $Ind_1n_7$[6.931] | $Ind_2n_7$[6.104] |
| $Ind_1n_8$[6.068] | $Ind_2n_8$[5.636] |
| $Ind_1n_9$[6.548] | $Ind_2n_9$[6.584] |
| $Ind_1n_{10}$[6.076] | $Ind_2n_{10}$[6.032] |
| $Ind_1n_{11}$[6.345] | $Ind_2n_{11}$[6.924] |
| $Ind_1n_{12}$[6.73] | $Ind_2n_{12}$[6.184] |
| $Ind_1n_{13}$[6.595] | $Ind_2n_{13}$[6.231] |
| $Ind_1n_{14}$[7.028] | $Ind_2n_{14}$[7.448] |
| $Ind_1n_{15}$[6.393] | $Ind_2n_{15}$[7.014] |
| $Ind_1n_{16}$[7.263] | $Ind_2n_{16}$[7.076] |
| $Ind_1n_{17}$[6.28] | $Ind_2n_{17}$[6.817] |
| $Ind_1n_{18}$[7.096] | $Ind_2n_{18}$[6.787] |
| $Ind_1n_{19}$[6.212] | $Ind_2n_{19}$[7.301] |
| $Ind_1n_{20}$[6.373] | $Ind_2n_{20}$[7.344] |
| $Ind_1n_{21}$[7.672] | $Ind_2n_{21}$[7.267] |
| $Ind_1n_{22}$[7.888] | $Ind_2n_{22}$[6.968] |
| $Ind_1n_{23}$[6.259] | $Ind_2n_{23}$[6.182] |
| $Ind_1n_{24}$[6.606] | $Ind_2n_{24}$[6.535] |
| $Ind_1n_{25}$[6.39] | $Ind_2n_{25}$[6.683] |
| $Ind_1n_{26}$[6.579] | $Ind_2n_{26}$[6.744] |
| $Ind_1n_{27}$[6.828] | $Ind_2n_{27}$[6.328] |
| $Ind_1n_{28}$[6.932] | $Ind_2n_{28}$[7.064] |
| $Ind_1n_{29}$[6.364] | $Ind_2n_{29}$[6.961] |
| $Ind_1n_{30}$[6.488] | $Ind_2n_{30}$[7.119] |
| $Ind_1n_{31}$[6.553] | $Ind_2n_{31}$[5.985] |
| $Ind_1n_{32}$[6.632] | $Ind_2n_{32}$[6.657] |
| $Ind_1n_{33}$[7.61] | $Ind_2n_{33}$[7.245] |
| $Ind_1n_{34}$[6.922] | $Ind_2n_{34}$[6.566] |
| $Ind_1n_{35}$[7.215] | $Ind_2n_{35}$[6.787] |
| $Ind_1n_{36}$[6.935] | $Ind_2n_{36}$[7.15] |
| $Ind_1n_{37}$[7.196] | $Ind_2n_{37}$[6.908] |
| $Ind_1n_{38}$[5.75] | $Ind_2n_{38}$[7.19] |
| $Ind_1n_{39}$[6.78] | $Ind_2n_{39}$[6.292] |
| $Ind_1n_{40}$[7.122] | $Ind_2n_{40}$[6.555] |
| $Ind_1n_{41}$[6.926] | $Ind_2n_{41}$[7.232] |

**Table 3** Results obtained by analyzing $IndSet_1$ and $IndSet_2$

| Type of analysis | $IndSet_1$ | $IndSet_2$ |
|---|---|---|
| Mean/SEM/[LowerCI,UpperCI] | 6.5486/0.1179/[6.310, 6.787] | 6.5874/0.1054/[6.374, 6.8] |
| SD/Variance | 0.7549/0.5699 | 0.675/0.4456 |
| CV/Interpretation | 11.53/rel-hom. | 10.2/rel-hom. |
| Sample size | 41 | 41 |
| Median/Min/Max | 6.595/3.752/7.888 | 6.744/4.159/7.448 |

$IndSet_1$ and $IndSet_2$. For the verification, the algorithm called *Effective Pairing Verification*, abbreviated as *II-EffPair* it is used. This verifies the existence of an indirect linear correlation. Because the data sets $IndSet_1$ and $IndSet_2$ were not normal, the algorithm indicates the calculus of $R$ using the nonparametric *Spearman Coefficient of Correlation*. As a result we obtain $R$=0.4261, which suggests the existence of a positive correlation. We considered appropriate to set the confidence interval for $R$ with the specific confidence level 95%. The 95% confidence interval of $R$ being [0.127, 0.654], with both bounds higher than *0* indicates the existence of a positive correlation. As a second verification, a statistical test was applied in order to verify if $R$ is statistically different from 0. This is formulated as the hypothesis that there exists a positive correlation, based on the fact that in our case $R > 0$. It was considered by the use of the test with a significance level $\alpha Corr = 0.05$. The obtained result of the test was $PvalCorr$=0.002, $PvalCorr < \alpha Corr$, which indicated the existence of the correlation ($R$ is significantly different from zero). Based on this, it could be formulated the conclusion that the pairing (matching) can be considered effective. This also allows the formulation of the conclusion that the problem-solving intelligence behavior of the multiagent systems is indirectly positively ($R > 0$) correlated.

Based on the verification of the pairing assumption, for the comparison and classification of the studied CMASs intelligence, the two-sample Wilcoxon test for paired data (matched pairs) with two-tails was applied. The significance level was considered $\alpha Class$=0.05. Some additional calculus details are: the *sum of all signed ranks* ($W$) =-1; the *sum of positive ranks* (T+) =430; the *sum of negative ranks* (T-)=-431; the *number of pairs*=41; the *degrees of freedom*=*n-1*=40. The obtained two-tailed *PvalClass* was *PvalClass*=0.99; $PvalClass > \alpha Class$ (see *Step5* in the *MeasApplInt* algorithm). Based on this condition, the decision is that $H0^{ie}$ can be accepted, which confirms that $COP^{i1}$ and $COP^{i2}$ intelligence is statistically the same. They solve problems with the same intelligence. From classification point of view $COP^{i1}$ and $COP^{i2}$ should be included in the same intelligence class. The numerical difference of the two Central Intelligence Tendency Indicators is the result of natural variability in the intelligence of the studied CMASs. By repeating the experimental evaluation, it could be obtained Central Intelligence Tendency Indicators with slightly different values, but from comparison point of view, the conclusion will be the same related to their classification in intelligence classes.

In the performed experimental setup, the two studied CMASs were chosen based on the fact that in researches presented in the scientific literature such systems are considered intelligent. This is the case of simple cooperating agents who based on the efficient and flexible cooperation has an emergent intelligence at the systems' level. In such a study, *HE* could be represented by a single human specialist that should take an easily decidable decision, related to the type of the intelligence and its effective measuring. The studied CMASs are the case of systems with a single type of problem-solving intelligence. The measuring criterion also is easy to establish based on the effective need related to the necessary problem solving requirement. As intelligence level measuring method it was chosen the best-solution found during the search for the solution process.

**Table 4** $IndSet_1$ and $IndSet_2$ data normality analysis results

| Normality test results | | |
|---|---|---|
| | K-S Normality test | |
| | $IndSet_1$ | $IndSet_2$ |
| K-S Statistics/PvalNorm/Passed | 0.141/≈0.039/No | 0.152/≈0.018/No |
| | S-W Normality test | |
| | $IndSet_1$ | $IndSet_2$ |
| S-W Statistics/PvalNorm/Passed | 0.88/≈0/No | 0.862/≈0/No |

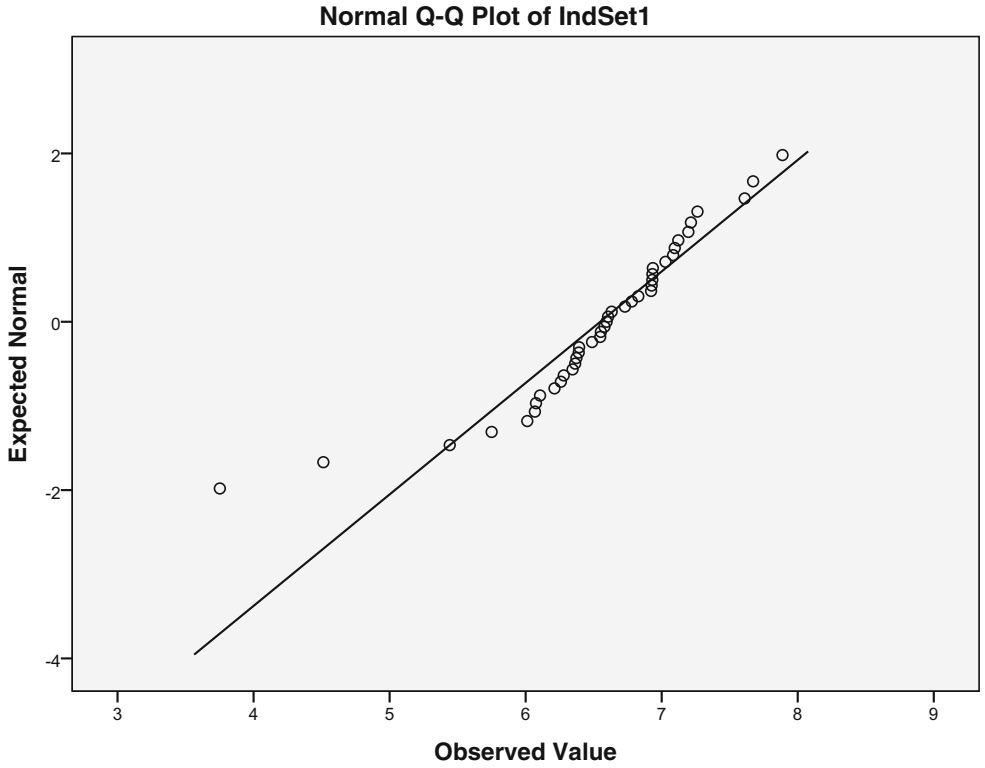



**Fig. 3** Normal Q-Q Plot of $IndSet_1$

This is an indicator of the quality of the solution. Another option that *HE* was possible to take consisting in intelligence measuring based on two components measure, the best-to-date solution found component and the problem-solving time component. This second possible decision influencing the obtained intelligence measure by applying

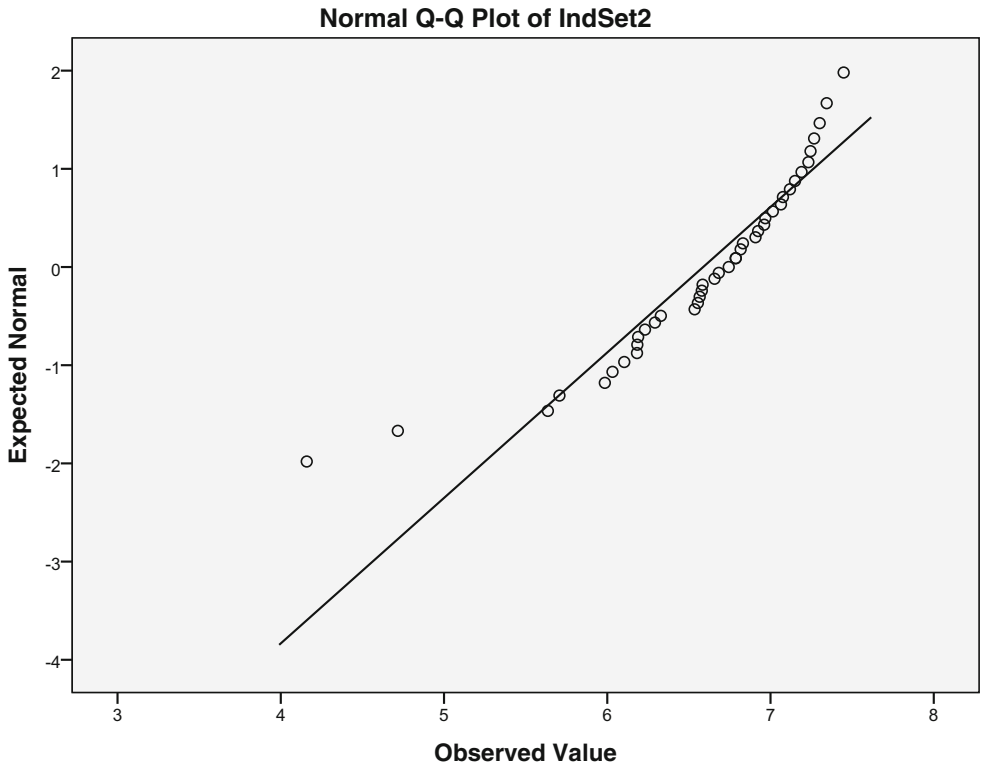



**Fig. 4** Normal Q-Q Plot of $IndSet_2$

the metric, being possible to give different evaluation result, but this is based on what effectively *HE* need (what is necessary effectively to be considered as the intelligence measure).

For evaluation and validation of the proposed metric, we conducted more experiments in that the CMASs were used in solving some computational hard problems. In another experimental study, we considered two CMASs denoted as $COP^{k1}$ and $COP^{k2}$. $COP^{k1}$ operated as an Ant Colony System (*ACS*) [20, 31, 32]. $COP^{k2}$ operated as a Best-Worst Ant System (*BWAS*) [25, 26, 91]. *ACS* was introduced by Dorigo and Gambardella [32]. *BWAS* was proposed in [25]. By applying the *MeasApplInt* metric to $COP^{k1}$ and $COP^{k2}$, it was obtained the result that the two studied CMASs cannot be included in the same class of intelligence. $COP^{k2}$ intelligence being lower than the $COP^{k1}$ intelligence. From applicative point of view, the obtained result should be interpreted that $COP^{k1}$ can solve more intelligently problems than $COP^{k2}$. The results of the performed experimental study are presented in the Supplementary Material attached to the paper.

### 4.3 Experimental comparison of the *MeasApplInt* and *MetrIntPair* metrics

In order to experimentally compare the *MeasApplInt* and *MetrIntPair* [45] metrics, on the obtained paired experimental problem-solving evaluation results presented in the previous subsection, it was applied the *MetrIntPair* metric. According to the *MetrIntPair* metric application assumption, it was verified the $IndSet_1$ and $IndSet_2$ normality using the requested One-Sample Kolmogorov Smirnov test [53] at the 0.05 significance level. The previously presented Table 4 presents the obtained test results. Based on the obtained results can be concluded that $IndSet_1$ and $IndSet_2$ does not passed the normality assumption. Based on this fact the *MetrIntPair* metric could not be directly applied. It was opted for a preprocessing step, consisting in the identification of outliers (extremely high and low) problem-solving intelligence. The using of Grubbs' test for outliers detection [4] is suggested when is expectable the normality after the elimination of extreme values. Table 2 presents the identified outliers, where "*" indicates an identified outlier, "#" indicates the pair of an identified outlier. The number that follows "*" specifies the application number of the test for outliers' detection (first application, second application and so one).

Let's make the assignment $IndSet_1$*= $IndSet_1$ and $IndSet_2$*= $IndSet_2$. First, it was applied the outliers' detection test on $IndSet_1$*. It identified $Ind_1n_1$[3.752] as an outlier, which was deleted from $IndSet_1$*, $IndSet_1$*=$IndSet_1^* - \{Ind_1n_1\}$; based on the pairing assumption it was deleted its pair $Ind_2n_1$[4.159] from $IndSet_2$*, $IndSet_2$*=$IndSet_2^* - \{Ind_2n_1\}$. Based on the fact that it was obtained an outlier, the outlier detection test was applied again on $IndSet_1$*, identifying $Ind_1n_4$[4.513] as a new outlier. It was deleted from $IndSet_1$*, $IndSet_1^* = IndSet_1^* - \{Ind_1n_4\}$; and deleted its pair $Ind_2n_4$[6.189] from $IndSet_2^*$, $IndSet_2^* = IndSet_2^* - \{Ind_2n_4\}$. Considering recursively, the outliers' detection test was applied again on $IndSet_1$*. This time, there was no any other outlier detected. Now, the application of the outlier's detection test on $IndSet_2$* began, identifying $Ind_2n_2$[4.717] as an outlier, $Ind_2n_2$ was eliminated from $IndSet_2$*, $IndSet_2^* = IndSet_2^* - \{Ind_2n_2\}$, based on the pairing property it was deleted $Ind_2n_2$ pair $Ind_1n_2$[5.442] from $IndSet_1$*, $IndSet_1^* = IndSet_1^* - \{Ind_1n_2\}$.

On the newly obtained $IndSet_1$* and $IndSet_2$*, obtained after the deletion of the extreme intelligence values from $IndSet_1$ and $IndSet_2$, it was applied the *K-S* normality test, with the obtained results presented in Table 5. The obtained results indicates that both $IndSet_1$* and $IndSet_2$* passed the normality assumption.

According to the next processing described in the *MetrIntPair* metric, in order to make the classification of the two studied CMASs, it was applied to the Paired Two-Sample T-test [30]. By using the two two-tailed test at the significance level $\alpha Mip$=0.05, the resulted *P-value* is 0.956. Based on the fact that the *P-value* $> \alpha Mip$ it can be concluded that the two studied systems have the same problem-solving intelligence.

As a next step for validation purposes, the pairing effectiveness was verified obtaining the correlation coefficient *R*=0.3498. For the verification, if *R* is significantly different from zero, a statistical significance test was applied, which returned the *PvalCorr*=0.0157. The result of $PvalCorr < \alpha Corr$, indicates that effective pairing resulted in a significant (with the 0.05 significance level considered) correlation (pairing/matching appears to be effective).

The results obtained by the *MeasApplInt* and *MetrIntPair* metrics on the same two studied and experimentally evalua-

**Table 5** Normality analysis results of $IndSet_1^*$ and $IndSet_2^*$

| Indicator | $IndSet_1^*$ | $IndSet_2^*$ |
|---|---|---|
| K-S Statistics | 0.086 | 0.091 |
| PvalNorm | ≈0.1 | ≈0.1 |
| Normality passed | Yes | Yes |

ted intelligent CMASs were the same. Both CMASs were identified as having the same problem-solving intelligence and is classified in the same class of intelligence.

## 5 Discussions

Based on comprehensive study of the scientific literature, we conclude that it is impossible to give a general accepted definition to the agent-based systems' (agents and cooperative multiagent systems) intelligence. Many definitions of artificial systems' intelligence presented in the scientific literature are based on some biological (nature) inspired considerations, like autonomous learning, self-adaptation and evolution. In a CMAS, the intelligence should be defined at the system's level. Many studies [46, 93] prove that even in very simple CMASs the intelligence could emerge at the system's level if the member agents cooperate efficiently and flexibly.

Most of the studies and researches focusing on the CMASs does not give any evaluation measure to the intelligence of the systems (how intelligent the investigated system is). We consider that there is not enough to give just a general definition to an agent-based system's intelligence. There are few results in the scientific literature related to this important research topic of measuring the machine intelligence. It is a real need to design effective metrics that allows a calculus of a CMAS intelligence and also offer the possibility to compare the intelligences of CMASs. We consider that the machine intelligence is based on the ability to solve difficult real-life problems. A designed intelligence metrics must be able to measure such an intelligence level.

In our research, we focused on the study of intelligence of CMASs. We measured the intelligence at the level of the whole cooperative multiagent system, not at individual/agent's level. Our metric, presented in the form of the algorithm *MeasApplInt*, is appropriate for CMASs, where the difficult problem-solving intelligence of a CMAS can be expressed as a single intelligence indicator. The human evaluator who wishes to measure and to compare the studied CMASs problem-solving intelligence should accomplish the selection of the most appropriate type of intelligence indicator. The type of intelligence indicator illustrates an aspect that the human evaluator considers that it characterizes the problem-solving intelligence. The designed metric can be used to detect intelligence type identified and selected by the human evaluator for the set of problems used in the experimental intelligence evaluations. This approach avoids the necessity to give a rigorous definition of the intelligence. In order to illustrate this affirmation, we mention the flying drone scenario presented in Section 3.1. In the considered scenario, the swarm is either formed by autonomous flying drones, by autonomous terrestrial drones (land vehicles), or, by completely autonomous flying and terrestrial drones. The swarm could even comprise hybrid drones that have some kind of autonomy, but it is not completely autonomous considering the fact that some coordinations could be made by humans.

In our study, we considered the calculus of the *CentrIntTen*, the indicator of the central intelligence tendency, as the median of the problem-solving intelligence evaluation's results. This was based on the requested robustness property of the metric (low sensitivity of the metric to the sample intelligence indicator data property). The median is a more robust indicator of the central intelligence tendency than the mean in case of data that are not normally distributed. A very high or very low numerical value influences the value of the median in a lesser extent than the mean's value.

In the paper [45] the *MetrIntPair* metric able to measure the machine intelligence of two CMASs was proposed. *MetrIntPair* is able also to compare the studied systems intelligence, and based on the comparison's result to make an accurate classification in classes of intelligence. *MeasApplInt* metric presented in this paper represents an improvement of the *MetrIntPair* metric from the robustness point of view. *MeasApplInt* considers the same principle of intelligence based on the real-life difficult problem-solving ability, as the *MetrIntPair* metric, and it is based on the same so-called pairwise/matched problem-solving intelligence evaluations. *MeasApplInt* is able to measure the machine intelligence, to compare and classify two intelligent systems. A CMAS could manifest its intelligence at a different level for different problems-solving. The designed *MeasApplInt* metric takes into consideration the variability in the intelligence of each of the compared CMASs.

The novelty of *MeasApplInt* is based on the difference in performed processing and analyses compared with *MetrIntPair*. *MetrIntPair* is based on the assumption of normality of the two intelligence indicators data sets, obtained as problem-solving intelligence evaluation results of the two studied CMASs. During the classification of the two studied multiagent systems, in order to verify if they belong to the same class of intelligence, *MetrIntPair* uses the parametric *Paired Two-Sample T-test* [30]. The t-statistic was introduced by William Gosset [10]. *MeasApplInt* does not request the passing of the normality assumption. This allows the principal advantage of robustness of *MeasApplInt* versus *MetrIntPair*. The robustness is based on the fact, that *MeasApplInt* for the classification of the two studied multiagent systems uses the nonparametric statistical *Wilcoxon signed rank test* for two paired data [24, 68, 69, 79, 87] and the *Mann-Whitney test* [35, 55]. The *Wilcoxon's test* is the nonparametric analog of the

*Paired Two-Sample T-test*. In *MeasApplInt* algorithm, the pairing/matching effectiveness is verified. The pairing effectiveness has the significance that there is an indirect correlation between the intelligence manifestations of the compared CMASs. For measuring the pairing effectiveness, the *II-EffPair* algorithm was proposed that is invoked by the *MeasApplInt* algorithm. If the pairing assumption passes, then the *Wilcoxon's* test is applied. If the pairing assumption fails, then the *Mann-Whitney test* is applied instead of *Wilcoxon's* test. The *Mann-Whitney test* is robust based on the fact that it does not request the passing of the assumption of normality, but it has less statistical power to detect differences than the *Wilcoxon test*. This is based on the fact that it is not designed for paired data. The power of a test of statistical significance is defined as the probability that it will reject a false null hypothesis. The power denoted in the following *Pow* can be calculated as $Pow = 1 - \beta$, where $\beta$ denotes the probability to make a type II error, that is, to reject the alternative hypothesis when this is true.

We would like to mention that if a data set is sampled from a Gaussian population, the nonparametric tests have less power, especially with small sample sizes. Based on this consideration, when the two intelligence indicator sample sizes are small and the samples pass the assumption of normality, we recommend the use of the *MetrIntPair* metric instead of the *MeasApplInt* metric.

*II-EffPair* algorithm can be applied even separately from the *MeasApplInt* metric for proving the indirectly correlated intelligent behavior in the problems solving of two studied CMASs. This represents in some extent the characterization of the problems solving behavior of the two studied CMASs. An indirect correlation indicates that the manifestations in the intelligence of the two studied intelligent systems is similar. In some cases, if for a problem, one of the systems' problem-solving intelligence is measured, and the measurements indicate low(high) intelligence, then it is expectable in some degree that the other system solves the same(or very similar) problem with similar low(high) intelligence also.

In the Section 4.3 an experimental comparison was performed related to the *MeasApplInt* and *MetrIntPair* metrics on the experimentally obtained problem solving intelligence evaluation results presented in Section 4.2. The results of the comparison and classification in the case of both metrics were the same. The presented comparative study illustrates the advantage of *MeasApplInt* versus the *MetrIntPair* consisting of its increased robustness. The comparative study presents a limitation of the *MetrIntPair* metric consisting in the fact that for its application it was necessary to identify and eliminate the extreme (low and high) intelligence indicator values. This does not affect the comparison and classification result but it altered the values of the obtained central intelligence tendencies of the two studied CMASs.

The *MeasApplInt* metric is universal, the applicability is not restricted to the studied cooperative multiagent systems architecture. It could be applied even for the intelligence evaluation of individual agents. We would like to emphasize that we use the universality notion with a different significance than is used in some researches, see for example [33], where the principle of metrics is used to measure the intelligence of artificial systems and biological life forms (humans for example) in the same extent.

## 6 Conclusions

Frequently, cooperative multiagent systems can solve difficult real-life problems in a more efficient and flexible way than the individual agents that operate in isolation. Intelligent cooperative multiagent systems are used for many real-life problem solving. Measuring the machine intelligence is very important in order to enable the differentiation between systems based on their problem-solving intelligence. There is no standardization, or even an universal assessment of what an intelligence metric should measure.

There are very few metrics able to make an effective quantitative measuring, also allowing the comparison of the systems and their classification into intelligence classes. In this paper, we proposed a novel universal metric called *MeasApplInt* that allows, in the same time, measuring, comparison and classification into intelligence classes of two studied intelligent CMASs. CMASs from the same intelligence class have the same real-life problem-solving intelligence. We consider that similarly to the humans and other intelligent life forms (apes, dolphins), the artificial intelligent systems have variability in intelligence. *MeasApplInt* takes into account the intelligence of the studied CMASs. The main advantage of *MeasApplInt* metric versus the *MetrIntPair* metric [45] consists in its robustness of the comparison of the intelligence of two studied intelligent systems. To decrease the necessary number of intelligence evaluations, and to allow the formulation of trustful conclusion, the principle of intelligence indicators data pairing/matching (making pairwise intelligence evaluations) was applied in our approach.

Based on a comprehensive study of the scientific literature we consider that our proposed metric is novel and it will represent the basis for intelligence comparison of cooperative multiagent systems in many future researches. As examples of possible applications, we mention the measuring and comparison of the intelligence of cooperative multiagent systems that are specialized in difficult problems-solving,

like: CMASs composed of flying drones with agent properties able to perform different missions like delivery of commanded products; CMASs composed of robotic agents able to fulfill variety of missions in physical environments, like search for different objects; CMASs composed of agents that are able to autonomously or semi-autonomously pilot specialized cars in transporting passengers.

We consider that some IBASs could have different types of intelligence. An IBAS could be more intelligent than another IBAS based on a specific type of intelligence, but for another type of intelligence the situation could be the vice versa. Our further research focuses on the proposal of a theory of multiple intelligences in intelligent computing systems. A sub-direction in this research consists in the automatic identification of the types of intelligence detained by intelligent systems.

**Acknowledgments** This work has been funded by the CHIST-ERA programme supported by the Future and Emerging Technologies (FET) programme of the European Union through the ERA-NET funding scheme under the grant agreements, title *SOON - Social Network of Machines*.

## References

1. Anthon A, Jannett TC (2007) Measuring machine intelligence of an agent-based distributed sensor network system. In: Elleithy K (ed) Advances and innovations in systems, computing sciences and software engineering. Springer, pp 531–535
2. Arif M, Illahi M, Karim A, Shamshirband S, Alam KA, Farid S, Iqbal S, Buang Z, Balas VE (2015) An architecture of agent-based multi-layer interactive e-learning and e-testing platform. Qual Quant 49(6):2435–2458
3. Arik S, Iantovics LB, Szilagyi SM (2017) OutIntSys - a novel method for the detection of the most intelligent cooperative multi-agent systems. In: Liu D et al (eds) 24th International conference on neural information processing, Guangzhou, China, November 14-18. Neural Information Processing, LNCS, 10637:31–40
4. Barnett V, Lewis T (1994) Outliers in statistical data, 3rd edn. Wiley, New York
5. Beni G, Wang J (1993) Swarm intelligence in cellular robotic systems. In: Dario P, Sandini G, Aebischer P (eds) Robots and biological systems: towards a new bionics? NATO ASI Series (Series F: Computer and Systems Sciences), vol 102. Springer, Berlin, pp 703–712
6. Bejar II, Whalen SJ (2003) Methods and systems for presentation and evaluation of constructed responses assessed by human evaluators, US Patent 6,526,258
7. Besold T, Hernandez-Orallo J, Schmid U (2015) Can machine intelligence be measured in the same way as human intelligence? Kunstl Intell 29(3):291–297
8. Boctor FF, Laporte G, Renaud J (2003) Heuristics for the traveling purchaser problem. Comput Oper Res 30:491–504
9. Bonett DG, Wright TA (2000) Sample size requirements for estimating Pearson, Kendall and Spearman correlations. Psychometrika 65:23–28
10. Box FJ (1987) Guinness, gosset, fisher, and small samples. Stat Sci 2(1):45–52
11. Brady SG, Fisher BL, Schultz TR, Ward PS (2014) The rise of army ants and their relatives: diversification of specialized predatory doryline ants. BMC Evol Biol 14:2–14
12. Bullnheimer B, Hartl RF, Strauss C (1999) A new rank based version of the ant system. A computational study. CEJOR 7(1):25–38
13. Chakraborty UK, Konar D, Roy S, Choudhury S (2016) Intelligent fuzzy spelling evaluator for e-Learning systems. Educ Inf Technol 21(1):171–184
14. Chakravarti IM, Laha RG, Roy J (1967) Handbook of methods of applied statistics, vol I. Wiley, New York, pp 392–394
15. Chliaoutakis A, Chalkiadakis G (2016) Agent-based modeling of ancient societies and their organization structure. Auton Agent Multi-Agent Syst 30(6):1072–1116
16. Coelho CGC, Abreu CG, Ramos RM, Mendes AHD, Teodoro G, Ralha CG (2016) MASE-BDI: Agent-based simulator for environmental land change with efficient and parallel auto-tuning. Appl Intell 45(3):904–922
17. Chmait N, Dowe DL, Green DG, Li YF, Insa-Cabrera J (2015) Measuring universal intelligence in agent-based systems using the anytime intelligence test. Technical Report, Monash University, Report Num, 2015/279
18. Chouhan SS, Niyogi R (2017) MAPJA: multi-agent planning with joint actions. Appl Intell 47(4):1044–1058
19. Colom R, Karama S, Jung RE, Haier RJ (2010) Human intelligence and brain networks. Dialogues Clin Neurosci 12(4):489–501
20. Colorni A, Dorigo M, Maniezzo V (1991) Distributed optimization by ant colonies. In: Actes de la première conférence européenne sur la vie artificielle. Paris, France, Elsevier Publishing, 134–142
21. Conley W (1988) Travelling salesman problem solved with simulation techniques. Int J Syst Sci 19(10):2115–2122
22. Conley W (1989) Two truck travelling salesman simulation. Int J Syst Sci 20(12):2495–2514
23. Conley W (1990) Multi-stage Monte Carlo optimization applied to a large travelling salesman problem. Int J Syst Sci 21(3):547–566
24. Conover WJ (1973) On methods of handling ties in the wilcoxon signed-rank test. J Am Stat Assoc 68(344):985–988
25. Cordon O, Herrera F, de Viana IF, Moreno L (2000) A new ACO model integrating evolutionary computation concepts: The Best-Worst ant system. In: Proceedings of ANTS'2000. From ant colonies to artificial ants: second international workshop on ant algorithms, Brussels, Belgium, September 7–9, 22–29
26. Cordon O, de Viana IF, Herrera F (2002) Analysis of the best-worst ant system and its variants on the QAP. In: Dorigo M, Di Caro G, Sampels M (eds) Ant algorithms, vol 2463. Springer, LNCS, Berlin, Heidelberg, pp 228–234
27. Crisan GC, Pintea CM, Palade V (2017) Emergency management using geographic information systems: application to the first Romanian traveling salesman problem instance. Knowl Inf Syst 50(1):265–285
28. Dantzig GB, Ramser JH (1959) The truck dispatching problem. Manag Sci 6:80–91
29. Dantzig G, Fulkerson D, Johnson S (1954) Solution of a large scale traveling salesman problem. Oper Res 2:393–410
30. David HA, Gunnink JL (1997) The paired t test under artificial pairing. Am Stat 51(1):9–12
31. Dorigo M (1992) Optimization, learning and natural algorithms. PhD thesis, Politecnico di Milano, Italy
32. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66
33. Dowe DL, Hernández-Orallo J (2014) How universal can an intelligence test be? Adapt Behavior Animals Animats Softw Agents Robots Adapt Syst Arch 22(1):51–69

34. Everitt B (1998) The cambridge dictionary of statistics Cambridge. Cambridge University Press, New York
35. Fay MP, Proschan MA (2010) Wilcoxon–mann–whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. Stat Surveys 4:1–39
36. Ferrucci D, Levas A, Bagchi S, Gondek D, Mueller ET (2013) Watson: beyond jeopardy! Artif Intell 199–200:93–105
37. Franklin D, Abrao A (2000) Measuring software agent's intelligence. In: Proceedings of international conference: advances in infrastructure for electronical business science and education on the internet. L'Aquila, Italy
38. Galton F (1886) Regression towards mediocrity in hereditary stature. J Anthropol Inst G B Irel 15:246–263
39. Grotschel M, Padberg MW (1978) On the symmetric travelling salesman problem: theory and computation. In: Henn R, Korte B, Oettli W (eds) Optimization and operations research. Lecture notes in economics and mathematical systems. vol 157, Springer, Berlin, pp 105–115
40. Hernandez-Orallo J, Dowe DL (2010) Measuring universal intelligence: towards an anytime intelligence test. Artif Intell 174(8):1508–1539
41. Hernández-Orallo J, Dowe DL, Hernández-Lloreda MV (2014) Universal psychometrics: measuring cognitive abilities in the machine kingdom. Cogn Syst Res 27:50–74
42. Hibbard B (2011) Measuring agent intelligence via hierarchies of environments. Artificial General Intelligence, Lecture Notes in Computer Science 6830:303–308
43. Hsieh FS (2017) A hybrid and scalable multi-agent approach for patient scheduling based on Petri net models. Appl Intell 7(4):1068–1086
44. Iantovics LB, Emmert-Streib F, Arik S (2017) Metrintmeas a novel metric for measuring the intelligence of a swarm of cooperating agents. Cogn Syst Res 45:17–29
45. Iantovics LB, Rotar C, Niazi AN (2018) Metrintpair-a novel accurate metric for the comparison of two cooperative multiagent systems intelligence based on paired intelligence measurements. Int J Intell Syst 33(3):463–486
46. Iantovics LB, Zamfirescu CB (2013) ERMS: an evolutionary reorganizing multiagent system, innovative computing. Inf Control 9(3):1171–1188
47. Iqbal S, Altaf W, Aslam M, Mahmood W, Khan MUG (2016) Application of intelligent agents in health-care: review. Artif Intell Rev 46(1):83–112
48. Johnson BR, Borowiec ML, Chiu JC, Lee EK, Atallah J, Ward PS (2013) Phylogenomics resolves evolutionary relationships among ants, bees, and wasps. Curr Biol 23(20):1–5
49. Jussila J, Vuori V, Okkonen J, Helander N (2017) Reliability and perceived value of sentiment analysis for twitter data. In: Kavoura A, Sakas D, Tomaras P (eds) Strategic innovative marketing. Springer proceedings in business and economics. Springer, Cham, pp 43–48
50. Kafali O, Yolum P (2016) PISAGOR: a proactive software agent for monitoring interactions. Knowl Inf Syst 47(1):215–239
51. Kwon H, Pack DJ (2012) A robust mobile target localization method for cooperative unmanned aerial vehicles using sensor fusion quality. J Intell Robot Syst 65(1):479–493
52. Leeuwen JV (ed) (1998) Handbook of theoretical computer science, vol A. Algorithms and complexity. Elsevier, Amsterdam
53. Lowry R Concepts & applications of inferential statistics. http://vassarstats.net/textbook
54. Mann PS (1995) Introductory statistics, 2nd edn. Wiley, New York
55. Mann HB, Whitney DR (1947) On a test of whether one of two random variables is stochastically larger than the other. Ann Math Stat 18(1):50–60
56. Marusteri M, Bacarea V (2010) Comparing groups for statistical differences: how to choose the right statistical test? Biochemia Medica 20(1):15–32
57. Merkle D, Middendorf M (2005) On solving permutation scheduling problems with ant colony optimization. Int J Syst Sci 36(5):255–266
58. Munteanu C, Rosa A (2004) Gray-scale image enhancement as an automatic process driven by evolution. IEEE Trans Syst Man Cybern B Cybern 34(2):1292–1298
59. Myers JL, Well AD (2003) Research design and statistical analysis, 2nd edn. Lawrence Erlbaum, Mahwah, p 508
60. Neisser U, Boodoo G, Bouchard TJ, Boykin AW, Brody N, Ceci SJ, Halpern DF, Loehlin JC, Perloff R, Sternberg RJ, Urbina S (1996) Intelligence: knowns and unknowns. Am Psychol 51(2):77–101
61. Newborn M (1997) Kasparov vs deep blue: computer chess comes of age. Springer, New York
62. Niazi M, Hussain A (2011) Agent-based computing from multi-agent systems to agent-based models: a visual survey. Scientometrics 89(2):479–499
63. Nick TG (2007) Descriptive statistics. Topics in biostatistics. Methods Mol Biol 404:33–52
64. Ouaarab A, Ahiod B, Yang XS (2014) Discrete cuckoo search algorithm for the travelling salesman problem. Neural Comput Appl 24:1659
65. Pearson K (1895) Notes on regression and inheritance in the case of two parents. Proc R Soc Lond 58:240–242
66. Pholdee N, Bureerat S (2016) Hybrid real-code ant colony optimisation for constrained mechanical design. Int J Syst Sci 47(2):474–491
67. Prakasam A, Savarimuthu N (2016) Metaheuristic algorithms and probabilistic behaviour: a comprehensive analysis of ant colony optimization and its variants. Artif Intell Rev 45(1):97–130
68. Pratt JW (1959) Remarks on zeros and ties in the Wilcoxon signed rank procedures. J Am Stat Assoc 54(287):655–667
69. Pratt JW, Gibbons JD (1981) Concepts of nonparametric theory. Springer, New York
70. Rosing MT (1999) 13C-depleted carbon microparticles in >3700-Ma sea-floor sedimentary rocks from West Greenland. Science 283(5402):674–676
71. Rouse WB, Sandra H (1983) Rouse analysis and classification of human error. IEEE Trans Syst Man Cybern SMC-13(4):539—549
72. Runkler TA (2005) Ant colony optimization of clustering models. Int J Int Syst 20:1233–1251
73. Schreiner K (2000) Measuring IS: toward a US standard. IEEE Intell Syst Their Appl 15(5):19–21
74. Sanghi P, Dowe DL (2003) A computer program capable of passing I.Q. tests. In: Slezak PP (ed) Proceedings of the joint international conference on cognitive science, 4th ICCS international conference on cognitive science and 7th ASCS Australasian society for cognitive science (ICCS/ASCS 2003). Sydney, NSW, Australia, pp 570–575
75. Sharkey AJC (2006) Robots, insects and swarm intelligence. Artif Intell Rev 26(4):255–268
76. Saska M, Vonasek V, Krajnik T, Preucil L (2014) Coordination and navigation of heterogeneous MAV–UGV formations localized by a 'hawk-eye'-like approach under a model predictive control scheme. Int J Robot Res 33(10):1393–1412
77. Shapiro SS, Wilk MB (1965) An analysis of variance test for normality. Biometrika 52(3-4):591–611
78. Sharpanskykh A, Haest R (2016) An agent-based model to study compliance with safety regulations at an airline ground service organization. Appl Intell 45(3):881–903
79. Siegel S (1956) Non-parametric statistics for the behavioral sciences. McGraw-Hill, New York, pp 75–83

80. Siorpaes K, Simperl E (2010) Human intelligence in the process of semantic content creation. World Wide Web 13(1-2):33–59
81. Song ZC, Ge YZ, Duan H, Qiu XG (2016) Agent-based simulation systems for emergency management. Int J Autom Comput 13(2):89–98
82. Stigler SM (1989) Francis galton's account of the invention of correlation. Stat Sci 4(2):73–79
83. Stutzle T, Hoos HH (1997) The MAX-MIN ant system and local search for the traveling salesman problem. In: Proceedings ICEC97. IEEE Press, Piscataway, pp 309–314
84. Stützle T, Hoos HH (2000) MAX MIN ant system. Futur Gener Comput Syst 16:889–914
85. Tokody D, Mezei IJ, Schuster G (2017) An overview of autonomous intelligent vehicle systems. In: Jármai K, Bolló B (eds) Vehicle and automotive engineering. Lecture notes in mechanical engineering, vol PartF12. Springer, pp 287–307
86. Turing AM (1950) Computing machinery and intelligence. Mind 59:433–460
87. Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1(6):80–83
88. Winklerova Z (2013) Maturity of the particle swarm as a metric for measuring the collective intelligence of the swarm. Advances in Swarm Intelligence, LNCS 7928:40–54
89. Won ZB, Do CB, Jeong YK, Han S (2002) Machine intelligence quotient: its measurements and applications. Fuzzy Sets Syst 127(1):3–16
90. Zarandi MHF, Hadavandi E, Turksen IB (2012) A hybrid fuzzy intelligent agent-based system for stock price prediction. Int J Intell Syst 27(11):947–969
91. Zhang Y, Wang H, Zhang Y, Chen Y (2011) Best-worst ant system. In: Proceedings of the 3rd international conference on advanced computer control (ICACC), pp 392–395
92. Yager RR (1997) Intelligent agents for World Wide Web advertising decisions. Int J Intell Syst 12(5):379–390
93. Yang K, Galis A, Guo X, Liu D (2003) rule-driven mobile intelligent agents for real-time configuration of IP networks, knowledge-based intelligent information and engineering systems. Lect Notes Comput Sci 2773:921–928




**László Barna Iantovics** received his B.Sc and M.Sc degree in Mathematics and Informatics at the Transylvania University of Brasov; a Ph.D. in Artificial Intelligence at Babes-Bolyai Univ. of Cluj-Napoca and finished a Postdoctoral Study in Artificial Intelligence at Alexandru Ioan Cuza University of Iasi. Dr. Iantovics is Associate Professor at University of Medicine, Pharmacy, Sciences and Technology of Targu Mures (UMFST). He is the Director of the Research Center "Advanced Computational Technologies" from UMFST. His principal research interests include the Intelligent Systems, Measuring the Machine Intelligence, Computational Intelligence, Biostatistics, and Artificial Intelligence applied for difficult problems solving in healthcare, topics on that he published dozens of papers and book chapters and contributed to more research projects as project director or researcher.

**László Kovács** is an associate professor with habilitation and the head of Institute of Information Sciences at University of Miskolc. He is an expert in knowledge engineering technology (ontology, semantic modelling, formal concept analysis) and natural language processing. He is or has been member of the programme committee of many conferences and workshops and he is a member of the editorial board for many journals in these areas. His extensive list of publications covers many scientific publication media in knowledge engineering, including the international conferences and journals. Moreover, he designed MSc and Bachelor study programs at University of Miskolc in Computer Science.

**Corina Rotar** is associate Professor at Sciences and Engineering Faculty from 1 Decembrie 1918 University of Alba Iulia. Her research activity focuses on bio-inspired computing, computational intelligence, and multi-objective optimization. In 15 years of activity, she performs teaching activity on several subjects as artificial intelligence, evolutionary computation, object-oriented programming, coordinates the scientific events, participates in research projects and directs the administrative activity of the Exact Science and Engineering Faculty. She is the author of more than 50 research articles and 2 books. She holds a Ph.D. from Babes-Bolyai University in Computer Science.

## Affiliations

**László Barna Iantovics[1] · László Kovács[2] · Corina Rotar[3]**

László Kovács
kovacs@iit.uni-miskolc.hu

Corina Rotar
crotar@uab.ro

[1] Informatics Department, University of Medicine, Pharmacy, Sciences and Technology of Targu Mures, Gh. Marinescu 38, Mures County, Targu Mures, 540139, Romania

[2] Information Technology Department, University of Miskolc, Miskolc, Hungary

[3] Computer Science Department, 1 Decembrie 1918 University, Alba Iulia, Romania

**Applied Intelligence manuscript No.**
(will be inserted by the editor)

# MeasApplInt - a novel intelligence metric for choosing the computing systems able to solve real-life problems with a high intelligence

## Appendix. Additional experimental evaluation of *MeasApplInt* metric

**L.B. Iantovics · L. Kovács · C. Rotar**

### Experimental evaluation of *MeasApplInt* metric for two CMASs

$COP^{i1}$ and $COP^{i2}$ cooperative multiagent systems studied in Subsection 4.2 proved that both of them belong to the same intelligence level. Based on this fact they were classified in the same intelligence class. From the viewpoint of practical applications, this result means that $COP^{i1}$ and $COP^{i2}$ are able to solve problems with the same intelligence. In this additional experimental study we present two CMASs whose problem-solving intelligence level is different and based on this fact they cannot be classified in the same intelligence class.

In the following, we present the experimental evaluation study of the proposed *MeasApplInt* metric to evaluate two CMASs denoted as $COP^{k1}$ and $COP^{k2}$. The cooperative problem solving by $COP^{k1}$ and $COP^{k2}$ is similar to the cooperative problem solving performed by the CMASs studied in Subsection 4.2. We take the sTSP optimisation task similar to the problem presented in Subsection 4.1. The human evaluator used the same problem-solving intelligence indicator measure as we have presented in the experimental study described in Subsection 4.2. It can be mentioned that lower intelligence indicator value means higher intelligence.

In $COP^{k1}$, we have implemented an Ant Colony optimization engine based on the Ant Colony System (ACS) [32, 20, 31]. Applications of ACS include among others: assembly sequence planning based on connector concept [94], solving single source capacitated facility location problem [95] and GPS positioning networks design [96]. $COP^{k2}$ involves Best-Worst Ant System (BWAS) [25, 26, 91] engine for the optimization. Applications of BWAS include: train operation daily schedule [97] and quadratic assignment problems [98]. The comparison tests were performed on maps of $nr$=50 cities. The parameter values for the studied CMASs where the same as in the experimental study presented in Subsection 4.2. *Repetitions*=1200; $\alpha$=1.738; $\beta$=2.085 and $\rho$=0.163.

**Table A.** Intelligence evaluation results of the $COP^{k1}$ and $COP^{k2}$

| $IndSet_A$ | $IndSet_B$ |
|---|---|
| $Ind_An_1$[5716] | $Ind_Bn_1$[8430] |
| $Ind_An_2$[7203] | $Ind_Bn_2$[9519] |
| $Ind_An_3$[6835] | $Ind_Bn_3$[9351] |
| $Ind_An_4$[7330] | $Ind_Bn_4$[9198] |
| $Ind_An_5$[3140] | $Ind_Bn_5$[5955] |
| $Ind_An_6$[6539] | $Ind_Bn_6$[9546] |
| $Ind_An_7$[6329] | $Ind_Bn_7$[8087] |
| $Ind_An_8$[6679] | $Ind_Bn_8$[8719] |
| $Ind_An_9$[6739] | $Ind_Bn_9$[9560] |
| $Ind_An_{10}$[6501] | $Ind_Bn_{10}$[10110] |
| $Ind_An_{11}$[6676] | $Ind_Bn_{11}$[8128] |
| $Ind_An_{12}$[6429] | $Ind_Bn_{12}$[9124] |
| $Ind_An_{13}$[6483] | $Ind_Bn_{13}$[9098] |
| $Ind_An_{14}$[6298] | $Ind_Bn_{14}$[9521] |
| $Ind_An_{15}$[7307] | $Ind_Bn_{15}$[10210] |
| $Ind_An_{16}$[4696] | $Ind_Bn_{16}$[7633] |
| $Ind_An_{17}$[6445] | $Ind_Bn_{17}$[9441] |
| $Ind_An_{18}$[7035] | $Ind_Bn_{18}$[9213] |
| $Ind_An_{19}$[6127] | $Ind_Bn_{19}$[9310] |
| $Ind_An_{20}$[6650] | $Ind_Bn_{20}$[9392] |
| $Ind_An_{21}$[7111] | $Ind_Bn_{21}$[10330] |
| $Ind_An_{22}$[6992] | $Ind_Bn_{22}$[10560] |
| $Ind_An_{23}$[6150] | $Ind_Bn_{23}$[8872] |

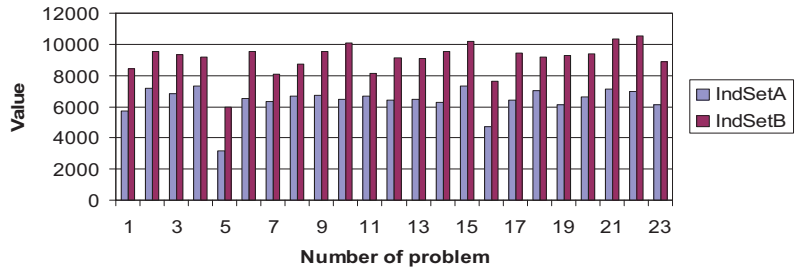



**Fig. A** Graphical representation of $IndSet_A$ and $IndSet_B$

Based on the specificity of the proposed metric there were performed paired experimental problem-solving intelligence evaluations for $|DifPr|$=23 problems. $IndSet_A$ denotes the experimental problem-solving intelligence evaluation results for $COP^{k1}$. $IndSet_B$ denotes the experimental problem-solving intelligence evaluation results for $COP^{k2}$. Table A and Figure A present the obtained $IndSet_A$ and $IndSet_B$ data sets.

Table B presents the results of the performed statistical characterization of $IndSet_A$ and $IndSet_B$ as indicated in the first step of the *Robust CMASs Intelligence Measuring, Comparison and Classification* algorithm. For the calculation of the confidence interval of the mean, we used the setting $CL$=95%. The selected $CV$ values of $IndSet_A$ and $IndSet_B$ indicate a moderate relative homogeneity. $CentrInt_A = Median(Ind_An_1, Ind_An_2, \ldots, Ind_An_k)$. $CentrInt_B$

**Table B.** Results obtained by analyzing $IndSet_A$ and $IndSet_B$

| **Type of analysis** | $\boldsymbol{IndSet_A}$ | $\boldsymbol{IndSet_B}$ |
|---|---|---|
| *Mean/SEM* | 6409.13/190.18 | 9100.3/206.86 |
| *[LowerCI,UpperCI]* | [6014.7, 6803.6] | [8671.3, 9529.3] |
| *SD/Variance* | 912.07/831871.7 | 992.05/984163.2 |
| *CV/Interpretation* | 14.2/rel-hom. | 10.9/rel-hom. |
| *\|DifPr\|/Median* | 23/6539 | 23/9310 |
| *Min/Max* | 3140/7330 | 5955/9310 |

**Table C.** $IndSet_A$ and $IndSet_B$ data normality analysis results

| ***Normality test results*** | ***K-S Normality test*** | |
|---|---|---|
| | $IndSet_A$ | $IndSet_B$ |
| *K-S Statistics/PvalNorm* | 0.248/0.0007 | 0.195/0.0238 |
| *Norm. Passed* | No | No |
| | ***S-W Normality test*** | |
| | $IndSet_A$ | $IndSet_B$ |
| *S-W Statistics/PvalNorm* | 0.745/≈0 | 0.885/0.013 |
| *Norm. Passed* | No | No |

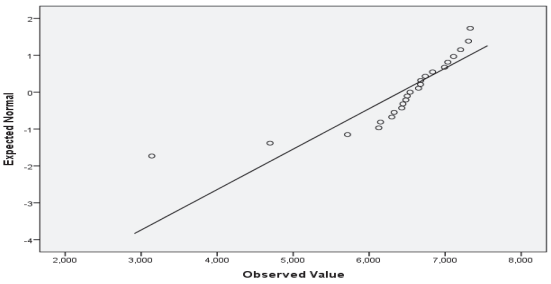



**Fig. B.** Q-Q Plot of $IndSet_A$

$=Median(Ind_B n_1, Ind_B n_2, \ldots, Ind_B n_k)$. $CentrInt_A$, $CentrInt_A$=6539, represents the calculated central intelligence tendency of $COP^{k1}$. $CentrInt_B$, $CentrInt_B$=9310, represents the calculated central intelligence tendency of $COP^{k2}$.

Table C presents the results of the performed normality tests *One-sample Kolmogorov-Smirnov test (K-S test)* and *Shapiro-Wilk test* (*S-W test*) applied to $IndSet_A$ and $IndSet_B$ data sets. Both of them were applied with the significance level $\alpha Norm$=0.05 considered the most appropriate in most of the cases. *PvalNorm* denotes the *P-value* of the performed normality test. Figures B and C presents the Q-Q Plot of $IndSet_A$ and $IndSet_B$ appropriate for the visual analysis of the normality. The obtained normality test results presented in Table C, and the visual representation of Figures B and C, shows that none of the intelligence indicator data sets $IndSet_A$ or $IndSet_B$ does not pass the normality assumption.

As a next step, it was applied the *Effective Pairing Verification* algorithm for the verification of the pairing effectiveness. Based on the fact that $IndSet_A$ and $IndSet_B$ were not normally distributed, according to the algorithm it was calculated the nonparametric Spearman coefficient of correlation denoted as

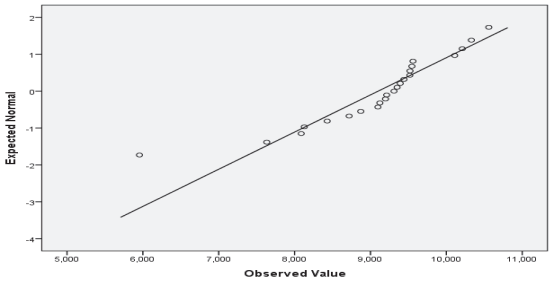



**Fig. C.** Q-Q Plot of $IndSet_B$

$R$. The $R$-value by $R$=0.574 suggests the existence of a positive correlation. During the verification, if $R$ is significantly different from zero a statistical significance test it was applied. In the tests, we set the significance level to $\alpha Corr$=0.05. The $PvalCorr$=0.0021, obtained as one-tailed P-value of the statistical test shows the existence of a positive correlation, which indicates an effective pairing (matching) of the obtained experimental problem-solving intelligence evaluation results of the studied CMASs. The conclusion is coming from the fact that $PvalCorr$ value is lower then $\alpha Corr$. Also, it was calculated the confidence interval of $R$ using the confidence level by 95% that in most of the cases is the most appropriate. The obtained confidence interval was by [0.1996, 0.8023]. The fact that both limits of the confidence interval are positive numbers, leads to the same conclusion as the previously mentioned significance test related to the effectiveness of the pairing.

According to the *Robust CMASs Intelligence Measuring, Comparison and Classification* algorithm for performing the comparison of the studied CMASs intelligence and their classification in intelligence classes it was applied the non-parametric two-sample Wilcoxon test for two paired samples with the significance level $\alpha Class$=0.05. As the test result, it was obtained $PvalClass \approx 0.0001$ (the obtained P-value of the Wilcoxon test). $PvalClass < \alpha Class$, indicates a statistically significant difference between the intelligence level of $COP^{k1}$ and $COP^{k2}$. However, $COP^{k1}$ and $COP^{k2}$ cannot be classified in the same intelligence class. $COP^{k1}$ having higher problem-solving intelligence than $COP^{k2}$ (conclusion formulated based on the fact that $CentrInt_A$ value is lower then the $CentrInt_B$ value; lower value indicating higher intelligence).

## References

20. Colorni A, Dorigo M, Maniezzo V (1991) Distributed Optimization by Ant Colonies, Actes de la premi'ere conference europeenne sur la vie artificielle, Paris, France, Elsevier Publishing, 134-142.
25. Cordon O, Herrera F, de Viana IF, Moreno L (2000) A New ACO Model Integrating Evolutionary Computation Concepts: The Best-Worst Ant System. Proc. of ANTS2000. From Ant Colonies to Artificial Ants: Second International Workshop on Ant Algorithms, Brussels, Belgium, September 79, (pp. 22-29)
26. Cordon O, de Viana IF, Herrera F (2002) Analysis of the Best-Worst Ant System and

Its Variants on the QAP. In Dorigo, M., Di Caro, G, Sampels, M. (Eds.), Ant Algorithms, Berlin, Heidelberg: Springer, LNCS, 2463:228–234
31. Dorigo M (1992) Optimization, Learning and Natural Algorithms, PhD thesis, Politecnico di Milano, Italy
32. Dorigo M, Gambardella LM (1997) Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1):53-66
91. Zhang Y, Wang H, Zhang Y, Chen Y (2011) Best-worst ant system. In Proceedings of the 3rd International Conference on Advanced Computer Control (ICACC), (pp. 392-395)
94. Tseng HE (2011) An Improved Ant Colony System for Assembly Sequence Planning Based on Connector Concept. In: Hu W. (eds) Electronics and Signal Processing. Lecture Notes in Electrical Engineering, vol 97. Springer, Berlin, Heidelberg, 881–888
95. Chen CH, Ting CJ (2006) Applying Multiple Ant Colony System to Solve Single Source Capacitated Facility Location Problem. In: Dorigo M., Gambardella L.M., Birattari M., Martinoli A., Poli R., Sttzle T. (eds) Ant Colony Optimization and Swarm Intelligence. ANTS 2006. Lecture Notes in Computer Science, vol 4150. Springer, Berlin, Heidelberg, 508–509
96. Saleh HA (2002) GPS Positioning Networks Design: An Application of the Ant Colony System. In: Dorigo M., Di Caro G., Sampels M. (eds) Ant Algorithms. ANTS 2002. Lecture Notes in Computer Science, vol 2463. Springer, Berlin, Heidelberg, 302–303.
97. Wu B, Xing Z, Zang Y (2018) Research on Train Operation Daily Schedule Based on Balanced Use. In: Jia L., Qin Y., Suo J., Feng J., Diao L., An M. (eds) Proceedings of the 3rd International Conference on Electrical and Information Technologies for Rail Transportation (EITRT) 2017. EITRT 2017. Lecture Notes in Electrical Engineering, vol 483. Springer, Singapore, 829-839
98. Cordon O, Fernandez de Viana I, Herrera F (2002) Analysis of the BestWorst Ant Systems and Its Variants on the QAP, Lecture Notes in Computer Science (Proc. III Int. Workshop on Ant Algorithms ANTS 2002), Berlin: Springer, no. 2463, pp. 228–234